Illustrating Time's Shadow

Supplements

Supplemental Shadows

This book addresses small indoor sundials of wood, glass, and PVC, as well as outside garden dials of glass, clay, tile, and common building materials. Less common dial features such as the inclined decliner and calendar or declination curves, are covered, as well as the astrolabe, other altitude dials and azimuth time keepers. This book uses empirical, geometric, trigonometric, CAD (computer aided design) both 2d and 3d, spreadsheet, procedural programming, tabular methods, and other techniques. Tables are provided.

Simon Wheaton-Smith

ILLUSTRATING TIME'S SHADOW

The Supplements

Supplemental Shadows enhances both
the book Illustrating Time's Shadow
as well as its associated Appendices

by Simon Wheaton-Smith

ISBN 978-0-9960026-1-5
Library of Congress Control Number: 2014904840
Simon Wheaton-Smith
www.illustratingshadows.com

June 12, 2018

ISBN 978-0-9960026-1-5
9780996002615

THE ILLUSTRATING SHADOWS COLLECTION

Illustrating Shadows provides several books or booklets:-

Simple Shadows	Build a horizontal dial for your location. Appropriate theory.
Cubic Shadows	Introducing a cube dial for your location. Appropriate theory.
Cutting Shadows	Paper cutouts for you to make sundials with.

Illustrating Times Shadow

the big book	Illustrating Times Shadow ~ Some 400 pages covering almost every aspect of dialing. Includes a short appendix.
Appendices	Illustrating Times Shadow ~ The Appendices ~ Some 180 pages of optional detailed appendix material.
Supplement	Supplemental Shadows ~ Material in the form of a series of articles, covers more on the kinds of time, declination confusion, other proofs for the vertical decliner, Saxon, scratch, and mass dials, Islamic prayer times (asr), dial furniture, and so on!

Programming Shadows A book discussing many programming languages, their systems and how to get them, many being free, and techniques for graphical depictions. This covers the modern languages, going back into the mists of time. Legacy languages include ALGOL, FORTRAN, the IBM 1401 Autocoder and SPS, the IBM 360 assembler, and Illustrating Shadows provides simulators for them, including the source code. Then C, PASCAL, BASIC, JAVA, Python, and the Lazarus system, as well as Octave, Euler, and Scilab. And of course DeltaCAD and its Basic variant, Python as in FreeCAD and Blender CAD systems, VBS and Java Script as in NanoCAD, programming TurboCAD (VBS and parametric script), and LISP as in the ProgeCAD system. And so on!

Illustrating Shadows provides a variety of software tools:-

CAD
DeltaCAD ~ macros for almost all dialing needs in BASIC
NanoCAD ~ dial macros written in VBS and Java Script
FreeCAD ~ dial macros written in Python
Powerdraw ~ dial macros in a Pascal subset
ProgeCAD ~ dial macros written in LISP
TurboCAD ~ dial macros written in VBS, and parametric part scripts also
Blender ~ dial macros written in Python

Languages Programs in the languages are discussed in Programming Shadows

Spreadsheets illustratingShadows.xls simpleShadows.xls cubicShadows.xls

Updates Check for general updates and corrections at:-

www.illustratingshadows.com/reference

or scan the QR code to the left which takes you there.

SUPPLEMENTAL SHADOWS MATERIAL

The supplements are really items that are a cross between what can be in appendices and what can be in the main book. They may also contain any corrections.

Simon Wheaton-Smith
www.illustratingshadows.com

KINDS OF TIME

Chapter 3 of Illustrating Time's Shadow discussed the planet Earth, and in so doing raised the question about time measurement, and of course that brings into the discussion the sundial, setting it, and reading it. There are three general topics to discuss:-

1. Sundial reading
2. Sundial calibration
3. Other hours and things
   a. unequal or biblical hours
   b. Babylonic hours
   c. Italic hours

FIRST SOME GENERAL THOUGHTS

Everyone knows about the time displayed on a clock. That time is called standard mean time. The word "mean" in this case relates to the "average" taken throughout the year. And in so doing, mean time for the clock or watch is divided days of equal amounts of time, and 24 equal hours in the average day, and each hour has 60 equal minutes, each has 60 equal seconds and so on. The day made up of equal numbers of equal hours is mostly fictitious, except for those few days when they match the reality of the sun. The word "standard" is used to create geographical areas where the clock time is the same, essential for commerce and travel. In essence:-

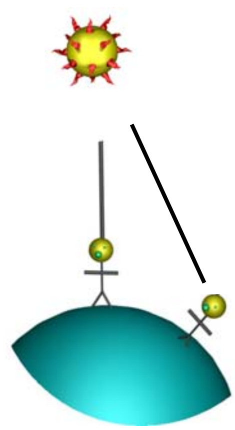
standard mean time =	the apparent time a simple sundial displays	+
	a correction for location	+
	a correction to make the sun time match mean time	

There is no intent to discuss sidereal time, which has a day being the Earth's rotation compared to a distant star.

SUNDIAL READING

Let us go back a bit farther and begin with the sun which apparently goes around the Earth. The reality is that planet Earth, and all the other planets, orbit around the sun. However for sundial purposes, it is easier to say the sun goes around the Earth.

The first measurement is then local sun time, called "Local Apparent Time", or "L.A.T" which should not be confused with the abbreviation "lat" for latitude. Local Apparent Time is the depiction of the sun usually in the form of a shadow, from a shadow casting device onto a surface. Local apparent time takes no notice of geographic location, nor of the day of the year. This was used for a long time, however compared to a clock, that time would not be correct for people east or west of our location, because the planet Earth is curved, so people east and west would see something different. See picture to the right.



When travel by stage coach was common, this worked out well. But as faster means of travel came along, it was a major nightmare when each town had its own "local apparent time"

That meant a geographic area (now called a time zone) would agree on a standard time, so a correction was needed for longitude, and thus developed the concept of standard time. For a small country such as England this worked well, for larger countries like the U.S.A. there was a need for time zones.

So, local apparent time became a standard time when adjusted for longitude.

An average day was used but since the day varies in length when based on the sun, another correction was needed. The difference between sun time and standard time was called the equation of time, or EOT.

Putting it all together:

standard mean time =	the local apparent time a simple sundial displays	+
	a correction for location based on longitude	+
	the equation of time, or EOT	

Local apparent time on a simple horizontal sundial is shown by hour lines whose angle from the north south line or local apparent noon is:-

local apparent time
hour line angle = arctan(tan(hour * 15) * sine(latitude))

The hours times 15 is based on the fact there are 16 degrees in an hour, which is based on a 24 hour day, and 360 degrees in the Earth's rotation. This formula is derived in many places in Illustrating Time's Shadow.

The longitude correction comes from knowing the longitude of the sundial, and the geographical longitude of the meridian that is the center of the time zone. For each degree away from that legal meridian, the correction is four minutes.

standard time correction
using longitude = (dial longitude – legal time meridian) * 4 in minutes

And converting standard time to standard "mean" time uses the equation of time. A simplistic table often used is:-

GENERIC EOT TABLE

	5th	15th	25th
JAN	5.1	9.0	12.0
FEB	13.9	14.2	13.4
MAR	11.9	9.3	6.3
APR	2.8	0.0	-2.3
MAY	-3.6	-4.0	-3.5
JUN	-2.0	-0.1	2.0
JLY	3.9	5.3	5.9
AUG	5.4	3.8	1.4
SEP	-2.2	-5.8	-9.4
OCT	-12.7	-15.1	-16.5
NOV	-16.5	-15.1	-12.5
DEC	-8.8	-4.5	0.2

There are several formula used to provide an accurate value, in particular the DeltaCAD macros and the spreadsheets provided with Illustrating Time's Shadow use either a 2 sine wave formula, a three sine wave formula, or a highly precise astronomically accurate formula.

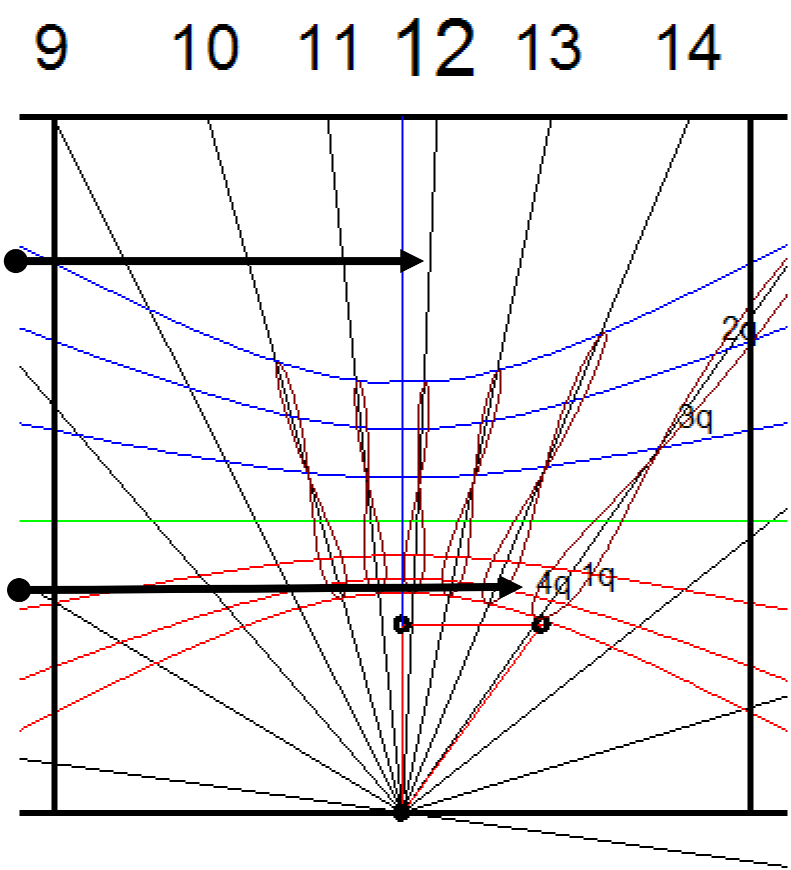
The only element missing from the recipe for standard mean time when offered a simple sundial is a political anomaly called daylight saving time, or summer time. In many places one hour is added to the clocks in the summer. SO the final formula is:-

legal time or standard time or clock time	=	LAT	+ {	EOT.corr eg Nov –15 eg Feb +12	+ west.long.corr – east.long.corr	+ 1 if summer }

That is all there is to it.

Many permanent sundials have the longitude correction built in, the way to tell is to look at the noon hour line. If it is north south on a horizontal dial, or vertical on a vertical dial, then there is no longitude correction. If the noon line is offset from north-south, or the vertical, then a longitude correction has been added.

Some sundials also have ingenious methods for providing the equation of time, or EOT, also. This involves a figure of eight shadow casting device, of a depicted figure of eight on each hour line. This figure of eight is the analemma. The DeltaCAD macros provided with Illustrating Time's Shadow provide analemma hourly dial plate depictions for most dial types and for some of them, a figure of eight bobbin like depiction when the hour lines have no analemma, but the shadow casting device does. Many sundials have an EOT chart such as on the preceding page.



The noon line on this northern hemisphere horizontal dial is offset to the east, so this dial was designed for a place eats of the legal meridian.

If the noon line was offset to the west then this dial would have been designed for a location west of the legal meridian.

The figure of eight analemma requires the observer to know the date. The markings 1q, 2q, 3q, and 4q indicate the season of the year.

And that is all there is to reading a sundial.

SUNDIAL CALIBRATION

Calibrating a sundial happens in two common situations. One situation happens when placing a new sundial into its final location. The other situation happens when a sundial is being created empirically. Both use the same very simple process,

What time is about to be calibrated or marked, and for simplicity, assume it is on the hour. What is the EOT value for the day, and the Illustrating Time's Shadow appendices have annual spreadsheets for each of the four years in the current era. Add the EOT value to the time to be calibrated or marked, and at that time, the shadow will be on the hour line.

For example, December 31, 2012 the EOT is:-   + 3m 26 seconds

So when the shadow is on the 9am line, it is 09:03:26

and conversely, at 09:03:26 then the shadow should be on the hour line.

It is that simple.

As a side note, if the tip of a shadow casting device were marked each day of the year at an exact clock time, say 12 noon, then the result would be a figure of eight analemma.

This can be a fun family activity.

OTHER HOURS AND THINGS

For the most part, sundials depict local apparent time, but quite a few have the longitude correction built in. As a rule, portable dials do not have the longitude correction built in, permanent installed dials often do.

But other hours are also depicted. Chapter 24 of Illustrating Time's Shadow addressed these in detail, however a summary here would make this article complete.

Length of day lines or curves

If the sun's declination is used to depict declination curves, or calendar curves, then they can be marked with a date (two dates except for the solstices) or they can be marked with the number of hours in the day. These are not hour lines as such, they are calendar or declination curves.

Unequal, temporary, or biblical hours

The unequal hours, temporary hours, or biblical hours are simply a given day's daylight divided into 12 equal parts. Those parts vary from day to day, shorter in winter, longer in summer, but are one hour long during the equinoxes. They are drawn by deriving the length of day at the winter solstice (for vertical dials) or summer solstice (for horizontal dials), dividing that duration by 12, and marking those times from noon on the solstice curve. At the equinox mark the times in 1 hour increments from noon. Connect the two points for each of the 12 “parts” and extend the lines. The result is a set of “hour” lines indicated by the nodus. Not much use these days, but of historical interest, and as such they can be found on some European dials.

Babylonian and Italian lines (see the DeltaCAD dial furniture macros for programming code)

Once a dial is built, and solstice and equinox lines and curves drawn, then all sorts of options exist for additional clutter (dial furniture) on the dial. Calendar information was often helpful, when to plant, sow, and reap in a then unhurried pace. In particular, some other interesting "hours" were used. Babylonian hours showed the time from sunrise. The Italian hours showed the time from sunset. In practice the Italian hours were massaged to show the number of hours until sunset, a figure helpful to the gardener or worker toiling in the fields.

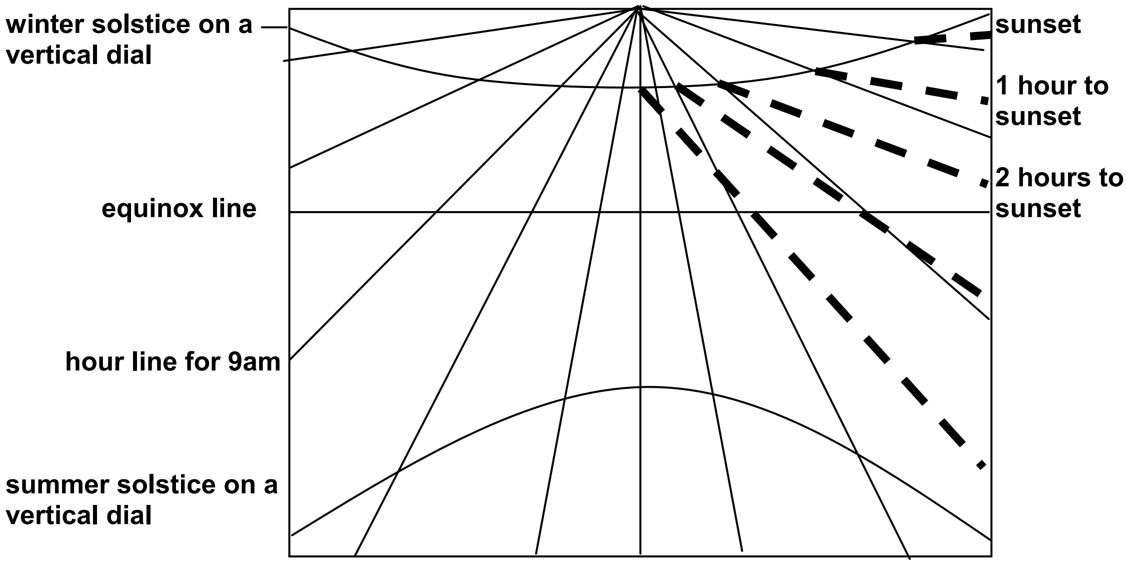
The technique for drawing Italian hour lines is simplicity itself. The time of sunset is noted for the solstices and the equinoxes. The appendices have approximate formulae for this calculation, and many online almanacs have readily available accurate times.

NOTE: If the dial is neither longitude nor EOT corrected, then use sunset times with no longitude or EOT correction. If the dial has longitude corrections built in, then use sunset times with a longitude but no EOT correction. If you use the spreadsheets on the web site, you can effect this by setting the reference longitude to the location's longitude, and the EOT can be set to zero by changing the EOT multiplier from 1 (normal) to 0 (no correction desired). Appendix 6 provides true (L.A.T.) sunrise, sunset, and day-length for many latitudes.

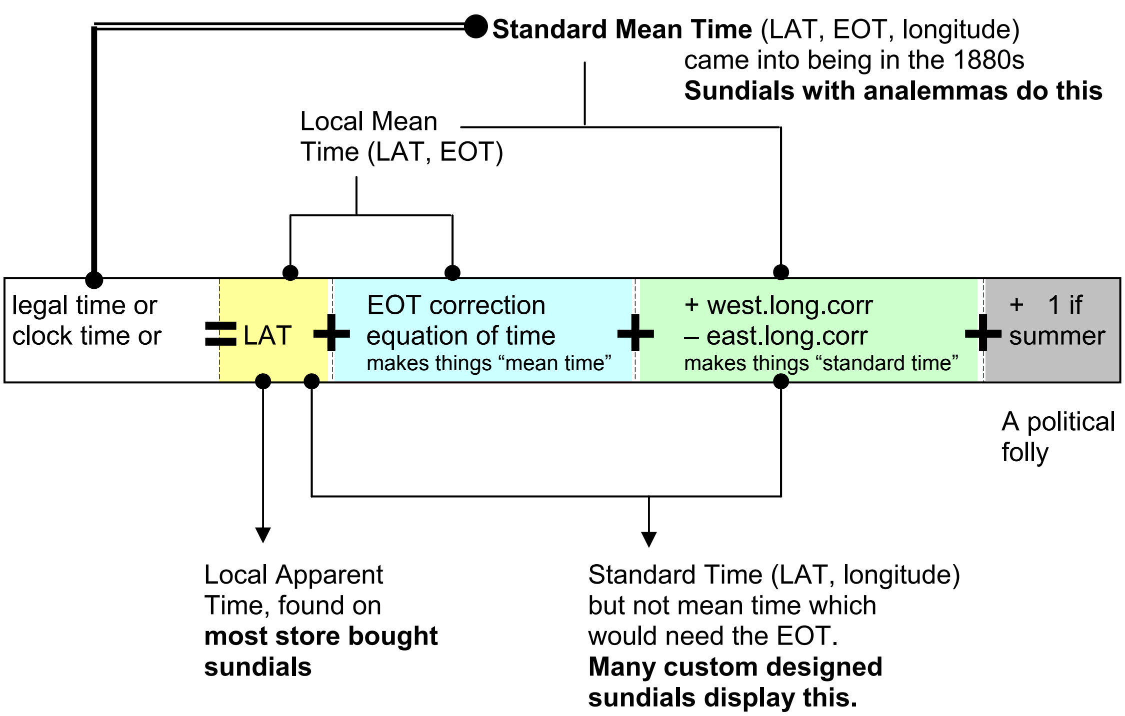
For a vertical dial, mark on the winter solstice calendar line the time when sun sets, and one hour before that, and two, and three, and so on. Then for the equinox perform the same operation (sunset is 6 pm L.A.T.), and similarly for the summer solstice if shown. Some dials may not have room for both solstice curves, so use just one solstice and the equinox. Simply connect the sunset time's dots on the solstice and equinox lines and draw a straight line. Mark that line as "sunset". Then connect the dots for one hour before sunset, and mark the resulting line "1 hour to sunset", and proceed for all the desired Italian hours. Take care with summer time adjustments when marking the hour points. Babylonian hours, while less useful, follow a similar process. The process is similar for horizontal or other dials.

CAUTION: Italian and Babylonian hours are latitude specific. Tilting an hour angle dial for a new latitude will correct the normal hours but not the Italian or Babylonian lines. While hour lines use the hour angle around the style, Italian and Babylonian lines are latitude and geometry based, detecting when and where the sun crosses the horizon. That geometry is latitude dependent as it uses the curvature of the Earth for a specific latitude and is thus not correctable by tilting.

A VERTICAL DIAL WITH SOME ITALIAN HOUR LINES



The various time definitions in perspective:-



EQUATION OF TIME

Equation of Time (EOT) **mm.ss** The mm.ss average of four years of astronomically accurate EOTs using 2010, 11, 12, and 13

	1	2	3	4	5	6	7	8	9	10	11	12
	Jan	**Feb**	**Mar**	**Apr**	**May**	**Jun**	**Jly**	**Aug**	**Sep**	**Oct**	**Nov**	**Dec**
1	3.15	13.31	12.26	3.59	-2.51	**-2.14**	3.47	6.22	0.09	-10.10	-16.27	-11.08
2	3.44	13.39	12.14	3.41	-2.58	**-2.05**	3.58	6.18	-0.10	-10.30	-16.28	-10.45
3	4.11	13.46	12.01	3.24	-3.05	**-1.55**	4.09	6.13	-0.29	-10.49	-16.29	-10.22
4	4.39	13.52	11.49	3.06	-3.11	**-1.45**	4.20	6.08	-0.49	-11.08	-16.29	-9.58
5	**5.06**	**13.58**	**11.35**	**2.49**	**-3.16**	**-1.35**	**4.31**	**6.03**	**-1.08**	**-11.26**	**-16.27**	**-9.34**
6	5.33	14.03	11.22	2.32	-3.21	**-1.24**	4.41	5.56	-1.28	-11.44	-16.26	-9.09
7	5.59	14.07	11.07	2.15	-3.25	**-1.13**	4.51	5.50	-1.49	-12.02	-16.23	-8.43
8	6.25	14.10	10.53	1.58	-3.29	**-1.02**	5.01	5.42	-2.09	-12.19	-16.19	-8.17
9	6.50	14.13	10.38	1.41	-3.32	**-0.50**	5.10	5.34	-2.30	-12.36	-16.15	-7.51
10	7.15	14.14	10.23	1.25	-3.35	**-0.39**	5.19	5.26	-2.51	-12.53	-16.10	-7.24
11	7.40	14.15	10.07	1.09	-3.37	**-0.27**	5.27	5.16	-3.12	-13.09	-16.04	-6.57
12	8.03	14.15	9.51	0.54	-3.38	**-0.14**	5.35	5.07	-3.33	-13.24	-15.57	-6.29
13	8.26	14.14	9.35	0.38	-3.39	**-0.02**	5.42	4.56	-3.54	-13.39	-15.49	-6.01
14	8.49	14.13	9.19	0.23	-3.40	**0.11**	5.49	4.46	-4.15	-13.54	-15.40	-5.32
15	**9.11**	**14.11**	**9.02**	**0.08**	**-3.39**	**0.23**	**5.56**	**4.34**	**-4.36**	**-14.08**	**-15.31**	**-5.04**
16	9.32	14.08	8.45	-0.06	-3.39	**0.36**	6.02	4.22	-4.58	-14.21	-15.20	-4.35
17	9.52	14.04	8.28	-0.20	-3.37	**0.49**	6.07	4.10	-5.19	-14.34	-15.09	-4.06
18	10.12	13.60	8.10	-0.34	-3.35	**1.02**	6.12	3.57	-5.41	-14.46	-14.57	-3.37
19	10.31	13.55	7.53	-0.47	-3.33	**1.15**	6.17	3.44	-6.02	-14.58	-14.44	-3.07
20	10.50	13.49	7.35	-0.60	-3.30	**1.28**	6.21	3.30	-6.23	-15.09	-14.30	-2.38
21	11.08	13.42	7.17	-1.12	-3.26	**1.41**	6.24	3.15	-6.45	-15.19	-14.15	-2.08
22	11.25	13.35	6.60	-1.24	-3.22	**1.54**	6.27	3.00	-7.06	-15.29	-14.00	-1.38
23	11.41	13.27	6.42	-1.36	-3.17	**2.07**	6.29	2.45	-7.27	-15.38	-13.44	-1.08
24	11.56	13.19	6.24	-1.47	-3.12	**2.20**	6.30	2.29	-7.48	-15.47	-13.27	-0.39
25	**12.11**	**13.10**	**6.05**	**-1.58**	**-3.07**	**2.33**	**6.31**	**2.13**	**-8.09**	**-15.54**	**-13.09**	**-0.09**
26	12.25	13.01	5.47	-2.08	-3.01	**2.46**	6.32	1.57	-8.30	-16.01	-12.51	0.21
27	12.38	12.50	5.29	-2.17	-2.54	**2.58**	6.32	1.40	-8.50	-16.07	-12.32	0.50
28	12.50	12.40	5.11	-2.27	-2.47	**3.11**	6.31	1.22	-9.11	-16.13	-12.12	1.20
29	13.01		4.53	-2.35	-2.39	**3.23**	6.29	1.04	-9.31	-16.17	-11.51	1.49
30	13.12		4.35	-2.43	-2.31	**3.35**	6.27	0.46	-9.51	-16.21	-11.30	2.18
31	13.22		4.17		-2.23		6.25	0.28		-16.24		2.47
												3.15

NOTE: The appendices have tables for the four years of the leap year cycle and in different centuries as well.

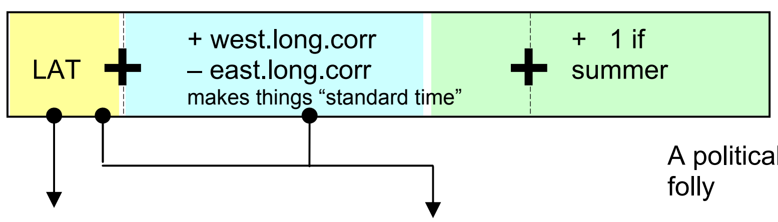
NOTE: This table comes from the “EOTandLONG” worksheet as a by-product, which is in the main spreadsheet: illustratingShadows.xls with no longitude consideration.

DIAL FURNITURE

A dial plate has on it hour lines, which is standard. Other information may exist on a dial plate, however, and this chapter addresses additional dial furniture. Furniture covered in Illustrating Time's Shadow or in the appendices will only be reviewed; other dial furniture will be expanded upon. But first, the progression of information on a dial plate in overview form might be helpful.

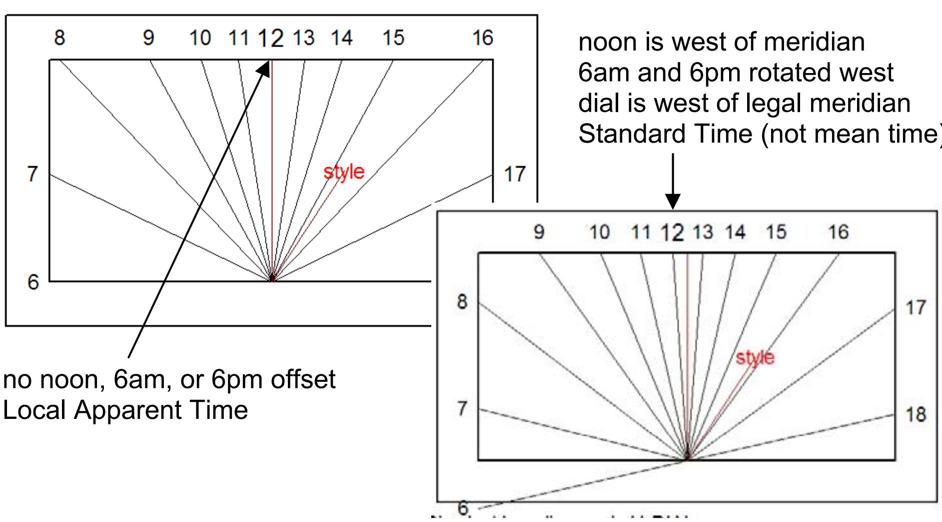
Hour lines — The most basic information is the hour line display. At its most basic form is shows Local Apparent Time (LAT). Such hour lines can be detected on a dial plate because noon is on the north south meridian for a horizontal dial, and vertical if a vertical dial.

Hour lines — The next development is the hour line display corrected for longitude, this then displays standard time. Standard time is local apparent time adjusted for the dial's longitude compared to the legal time meridian. This is not "standard mean time", since mean time requires an adjustment depending on the date, that adjustment comes from the equation of time, or EOT. Standard time lines can be detected because the noon line will be offset to the west if the dial is west of the legal meridian, and offset to the east if the dial is east of the meridian. This is true for both the north and south hemispheres. Additionally the 6am and 6pm lines will not be aligned with the east-west line, and may not be at 90 degrees to the noon line either. And this applies to horizontal dials as well as true north:south facing vertical dials.



Local Apparent Time, found on **most store bought sundials**

Standard Time (LAT, longitude) but not mean time which would need the EOT. **Many custom designed sundials display this.**



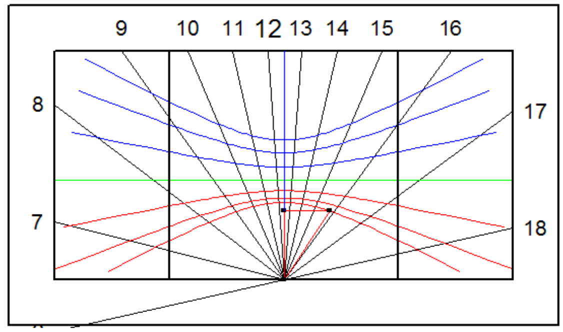
noon is west of meridian
6am and 6pm rotated west
dial is west of legal meridian
Standard Time (not mean time)

no noon, 6am, or 6pm offset
Local Apparent Time

The first example, Local Apparent Time or LAT is what is typical of store bought dials. However, portable dials are often not corrected for longitude because being portable they may be moved. Yes, moving them may also mean a latitude change; however that is fixed by tilting the dial towards the north if the dial moves south, or, tilting the dial to the south if the dial moves to the north.

Calendars

The next item of dial furniture as far as usage goes is often the declination curves, or calendar curves.



The equinox is a straight line; the solstices and any additional curves are hyperbolic curves.

Each curve represents a solar declination.

Common declinations used are:-

GOOD DECLINATIONS FOR REASONABLE SYMMETRY			
23.44 Jun, Dec	20 (also 19 or 18) Jan, May, Jly, Nov	10 (also 11 or 12) Feb, Apr, Aug, Oct	0 Mar, Sep

When selecting declinations other than for the solstices (23.44) and the equinoxes (0), the issue arises of symmetry. Declinations of plus and minus 20 as well as plus or minus 10 work reasonable well, however 19 and 11, or 18 and 12 also work. This is a case of “yer pays yer money, yer gets yer choice”.

There are often a total of seven calendar or declination curves. Except for the solstices, ambiguous dates exist.

	Jan	Feb	Mar	Apr	May	
Solstice~Dec			Equinox			Jun~Solstice
	Nov	Oct	Sep	Aug	Jly	

Chapter 23 of Illustrating Time’s Shadow covers the topic of declination curves. The simplest formula for solar declination is:

declination = dec = 23.45*sin((0.9678(j-80)))

and more complex ones exist, as covered in appendix 8 of the appendices.

Length of Day

The declination curves may be indicative of a date, often around the 21st of the month, however, each declination provides another item of data, namely the length of day. The length of day is dependent on both the solar declination, as well as the latitude.

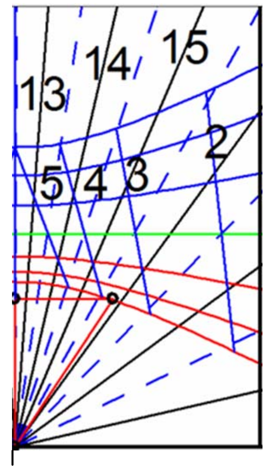
Chapter 24 of Illustrating Time’s Shadow covers the topic of length of day curves.

At the equinoxes, the length of day is 12 hours regardless of latitude, at the solstices the length can be anything from 0 to 24 hours (at the poles), and of course at the equator, all days are 12 hours in length.

Italian

The Italian line is an hour line that marks the number of hours since the previous sunset. Most people do not care how long it was since the previous sunset, most normal people care about how log until the next sunset, so how you choose to mark them is your affair. One common method is to mark Italian lines as the hours until sunset.

The method of drafting Italian hour lines is to establish the time of sunset during the winter solstice for a vertical dial, or the summer solstice for a horizontal dial. This is not a hard and fast rule, it is one of practicality. Then on the equinoxes, sunset is at 6pm Local Apparent Time. Then back off by one hour, and get the times for one hour before sunset, and repeat the process.

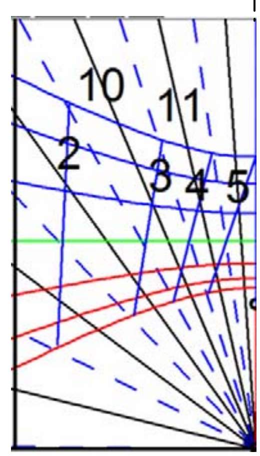


The picture to the right shows this for latitude 33.5 north, Phoenix, AZ. If the dial has no longitude correction, the formula for sunset is used as is. If the dial is longitude corrected, then that correction must be considered. The formula does not use the equation of time. So if you use an almanac rather than the formula for the time of sunset, you must adjust that time accordingly. The local hour angle for sunset is:

lhaRiseSet = arccos(tan(lat) * tan(dec))

Babylonian

The Babylonian hour line provides the time since sunrise. The process is substantially the same as for sunset.



Unequal, Temporary, or Biblical hours

The unequal hours, temporary hours, or biblical hours are simply a given day's daylight (see earlier) divided into 12 equal parts. Those parts vary from day to day, shorter in winter, longer in summer, one hour during the equinoxes. They are drawn by deriving the length of day at the winter solstice (for vertical dials) or summer solstice (for horizontal dials), dividing that duration by 12, and marking those times from noon on the solstice curve. At the equinox mark the times in 1 hour increments from noon. Connect the two points for each of the 12 "parts" and extend the lines. The result is a set of "hour" lines indicated by the nodus. Not much use these days, but of historical interest, and as such they can be found on some European dials.

Chapter 24 of Illustrating Time's Shadow covers the topic of both Italian and Babylonian hours as well as the unequal hours that have fallen into disuse.

Logically, the correction for mean time would follow next. Many dials do not build this into the gnomon or dial plate, they may use a table or graph instead. And many ignore the equation of time, which is one reason why people think sundials are inaccurate.

Regardless, to correct the dial reading to “mean time”, which means the average of the time slots throughout the year, the equation of time must be incorporated somehow. There are several ways of doing this.

Mean Time

The simplest way to establish mean time, or clock time, is to have an equation of time, or EOT, graph or table.

GENERIC EOT TABLE

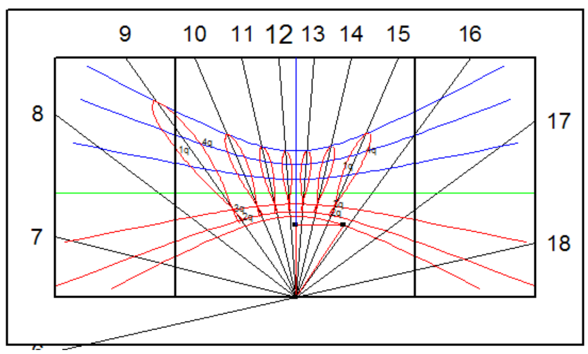
	5th	15th	25th
JAN	5.1	9.0	12.0
FEB	13.9	14.2	13.4
MAR	11.9	9.3	6.3
APR	2.8	0.0	-2.3
MAY	-3.6	-4.0	-3.5
JUN	-2.0	-0.1	2.0
JLY	3.9	5.3	5.9
AUG	5.4	3.8	1.4
SEP	-2.2	-5.8	-9.4
OCT	-12.7	-15.1	-16.5
NOV	-16.5	-15.1	-12.5
DEC	-8.8	-4.5	0.2

Some dials have the longitude correction built into the EOT table, this might be the case for a large dial in a park that had no original longitude adjustment designed. The master spreadsheet:-

illustratingShadows.xls provides this.

The second method of establishing mean time is to use an EOT graph, some of which are very interesting. This is the graphical version of the table.

The third method is to include the EOT o the dial plate along with the lines, and the fourth method is to design a bobbin for the gnomon’s style. In the first case, the dial plate has the EOT knowledge, and it can look quite busy. This is shown to the right.



In the second case the gnomon has the EOT knowledge, but building such a gnomon can be complex. In both cases, the season must be known.

Chapter 25 of Illustrating Time’s Shadow covers the topic of the analemma on both the dial plate and the gnomon.

What has been covered so far addresses the most common items of dial furniture.

LAT ~ Local Apparent Time

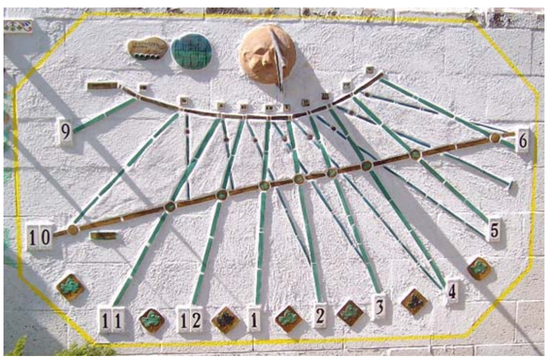
Longitude correction → Standard Time → EOT correction → Standard Mean Time

EOT correction → Local Mean Time → Longitude correction → Standard Mean Time

However, additional dial furniture can be added. As can be seen, a dial with hour lines (local or standard, or, no longitude or with a longitude correction) is fairly un-cluttered. Adding the declination curves, otherwise known as calendar curves, or length of day curves, begins to ad to the clutter. Adding mean time information will clutter the dial plate if the analemma is used, less so if a table or graph is employed.

To the left, a horizontal dial plate with standard time only, no declination curves, but with an equation of time chart.

To the right, a horizontal dial plate with standard time, as well as Babylonian and Italian lines.

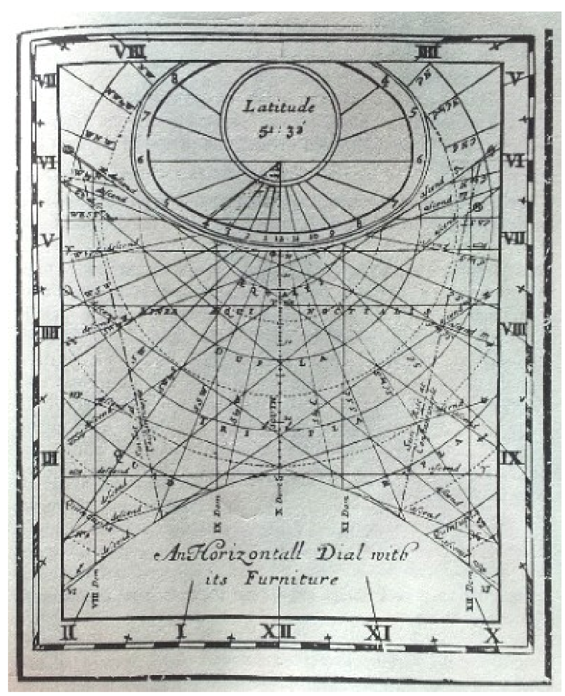


To the left a large vertical declining dial dial plate with standard time, as well as Italian lines, but with summer time marked at the lower part of the dial plate, which is where the shadow falls in the summer.

The dial plate gets cluttered easily. One solution was to have me than one dial plate, and when combined into a single instrument, it was termed “a compendium”.

Could there possibly be more dial furniture one might ask. The answer would be yes, of course, someone has to make a living, and work expands to fill the time.

To the right is a well known dial plate depiction, and as you can see, the word “clutter” is very appropriate!



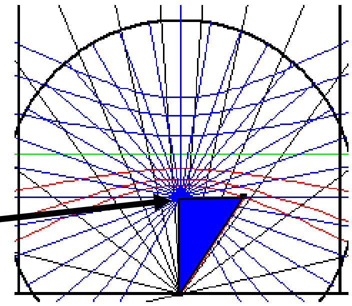
Azimuth

The azimuth of the sun is its angle from true north or south. For a horizontal dial, this would be a series of radials radiating from a point vertically below the nodus. For a vertical dial this would be a series of vertical lines paralleling the noon line, and starting from noon local apparent time. For the vertical dial, the line would have an "X" component equal to:-

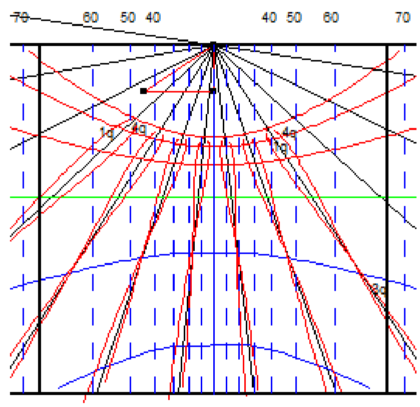
$$X = \text{gnomon linear height} * \tan(\text{azimuth})$$

A horizontal dial, longitude corrected (standard time but not standard mean time), with declination curves for calendar information, also has the radials for azimuth.

azimuth radials radiate from a point vertically below the nodus



A vertical dial, longitude corrected but with azimuth lines is shown to the right.

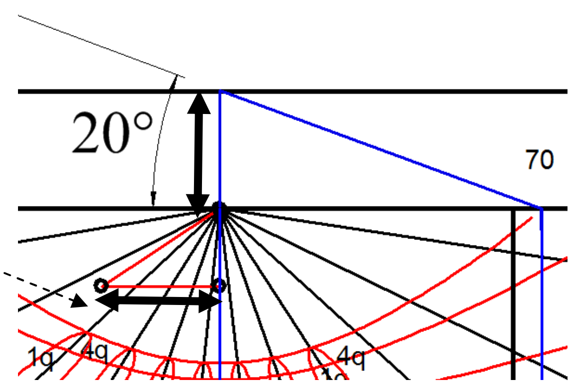


As can be seen, the azimuth radials on a horizontal dial place are rather busy, the ones on a vertical dial plate less so.

The process for the horizontal dial is very simple, and for a vertical dial, the trigonometry or geometry is also simple.

gnomon linear height is from the nodus to the dial plate

a common mistake is to use the dial center to the base of the nodus (sub style line)



Either way, azimuth information does add to the clutter on the dial plate.

Altitude

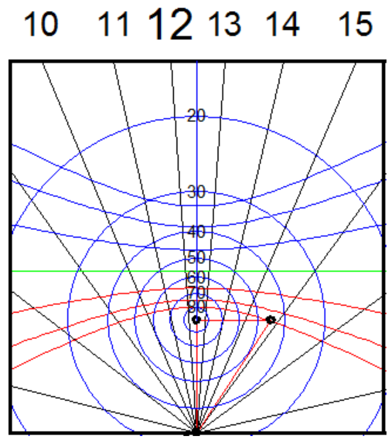
The altitude of the sun is the angle the sun is above the horizon, and except for noon, two times in the day will share the same altitude.

For a horizontal dial, this consists of a series of circles, each circle is a different altitude.

The radius of the circle is

R = gnomon linear height * tan(90-altitude)

and the altitude circles are drawn immediately below the nodus, on the dial plate.



For a vertical dial the altitude is a hyperbolic curve connecting the series of X,Y pairs for the desired altitude at each azimuth. For each azimuth point, its “X” ordinate is:-

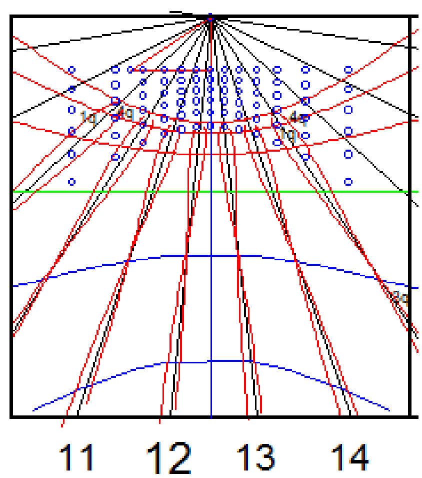
```
xxx = glh * Tan(azimuth)
```

But the “Y” ordinate is a bit more involved. If the azimuth is 0 then it is one simple formula, if the azimuth is non zero then the formula must consider the azimuth also:-

```
' xxx is the x value for this azimuth
yyy = glh * Tan(altitude)

If zzz > 0 Then
        yyy = xxx * Tan(altitude) / Sin(zzz)
End If
```

The test for an azimuth of 0 is required to avoid the division by 0.

The altitudes are shown to the right as a series of small circles, those that form a horizontal line are the altitude 0, and in the case to the right, altitude is shown for every ten degrees.



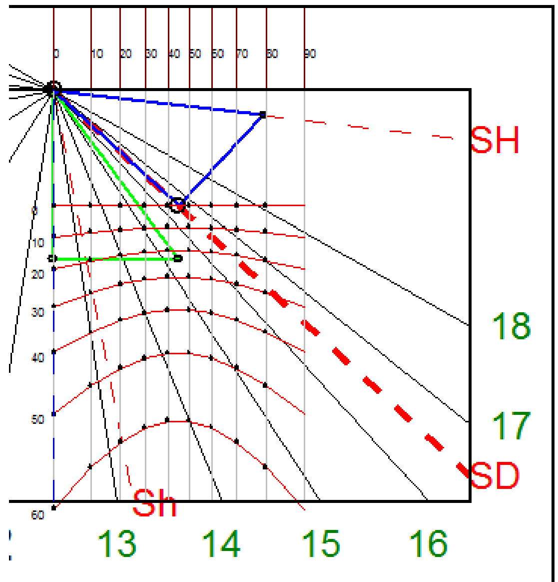
Two DeltaCAD macros provide dial furniture options. Choice 3 of the horizontal dial DeltaCAD macro can draw altitude circles and azimuth radials, and also allows the calendar curves and hour lines to be turned off.

Choice 7 of the vertical dial DeltaCAD macro allows azimuth lines and altitude points to be drawn.

AZIMUTH AND ALTITUDE ON A VERTICAL DECLINER

- Depicting the azimuth as vertical lines on a vertical decliner

- Depicting the altitude as curves on a vertical decliner



The formulae and the formula derivation for this process as well as the methods used to rotate the gnomon on a dial plate, using DeltaCAD macros, is covered in the book:-

Programming Shadows

The book of over 200 pages is available on the Illustrating Shadows web site and can be downloaded when you download the main book "Illustrating Time's Shadow". This book was completely revised in February of 2013.

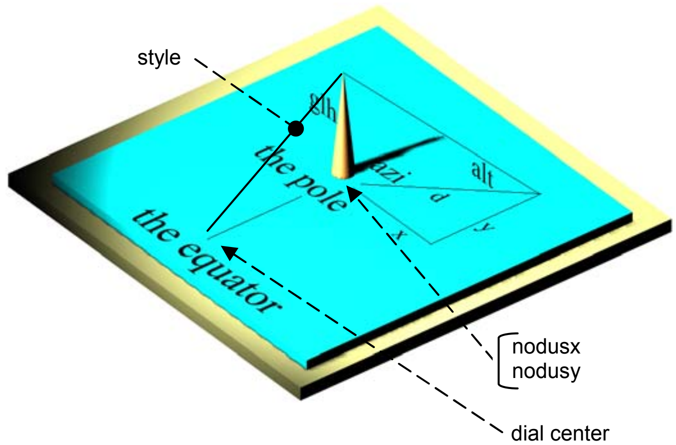
Using the altitude and azimuth approach, the next few pages show a simple approach to drafting declination curves, or calendar curves, or day length curves. While the algorithms are for a CAD program, shown in three programming languages, they can be used easily for manual drafting, and are well suited to using a spread sheet or a calculator.

Chapter 23 of Illustrating Time's Shadow covers several methods for drafting these curves. For the horizontal and vertical dials, that chapter focused on completeness. The following pages are simpler but not as extensive as the same section in the main book, and are very practical for any CAD program. If you had any difficulty with the chapter 23 treatment of the horizontal and vertical dials, then this short section should prove invaluable.

Declination lines using trigonometry.

Horizontal dial ~ calendar or declination curve logic

The simplest approach for a horizontal dial is to select a solar declination such as 23.44, –23.44, or 0, and run the hours from morning to afternoon. Running the hours means using LAT (local apparent time), and for each time increment, derive the sun's azimuth and altitude. That part of the process is the same for the vertical, horizontal, and vertical declining dial. Assume "glh" is the gnomon linear height:-



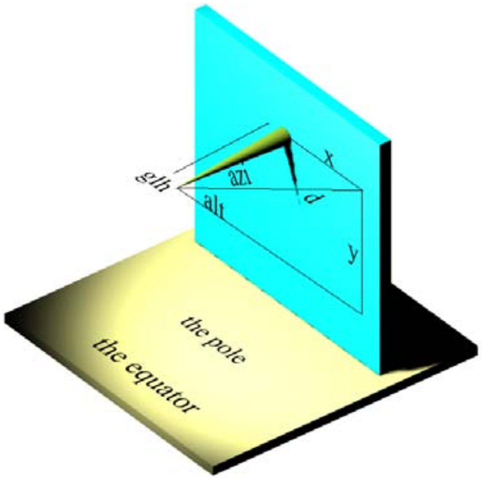
For the horizontal dial, distance "d" is:- d = glh / tan(altitude)
thus x = d * sin(azimuth)
and y = d * cos(azimuth)

The very first calculation is the first point of a declination or calendar curve, and thus no line or curve segment is drawn, but the "x,y" coordinates are saved. The hour is bumped up to the next time and a new "x,y" coordinate calculated, and this time a line segment is used because we have the previous "x,y" pair. This is repeated until the end of the time frame. This process can be repeated for all desired declinations. The "x,y" coordinates are relocated by the coordinates of the base of the nodus, "NODUSY, NODUSY", and not from dial center. Some logic is shown below, this is written in Java Script as used in the free NanoCAD program:-

```
/*#######################################################################################
' #  MAIN LOOP   #   declination in this example is –23.44   (winter solstice)         #
' #              #   makeLine is a function to draw a line segment, see "functions.js"  #
' #######################################################################################
*/

/* this uses NODUSX,Y from the gnomon drawing earlier */
wx = 9999                                 /* these, when 0, tell the calendar */
wy = 9999                                 /* line draw to draw nothing */

for (hr=7;  hr < 17;  hr=hr+0.25) {
   /* get an altitude and azimuth for this one hour at this SOLAR declination */
   al = altitude( hr, lat, -23.44 )      /* degrees altitude */
   d  = glh / Math.tan(al*2*3.1416/360)  /* radial distance: nodus base to cal curve */
   az =  azimuth( hr, lat, -23.44 )      /* degrees azimuth */
   xxx=  d * Math.sin(2*3.1416*az/360)
   yyy=  d * Math.cos(2*3.1416*az/360)
   if (  wx<9999 && wy<9999 ) {
         if ( Math.abs(nodusx+wx)<150 && Math.abs(nodusx+xxx)<150 ) {
            if ( Math.abs(nodusx+wx)<130 && Math.abs(nodusx+xxx)<130 ) {
               // above code keeps the curve within reasonable bounds
               makeLine(  nodusx + wx,nodusy + wy   ,   nodusx + xxx, nodusy + yyy  )
            }
         }
   }
   wx = xxx
   wy = yyy
}
```

The azimuth and altitude formula are the standard formulae, see the appendices.

Vertical dial ~ calendar or declination curve logic



As for the horizontal dial, the approach for a vertical dial is to select a solar declination and run the hours from morning to afternoon. Running the hours means using LAT (local apparent time), and for each time increment, derive the sun's azimuth and altitude. That part of the process is the same for the vertical, horizontal, and vertical declining dial. Assume "glh" is the gnomon linear height:-

For the vertical dial:- x = glh * tan(azimuth)

and, distance "d" is:- d = glh / cos(azimuth)

which is needed for the "y" ordinate:-

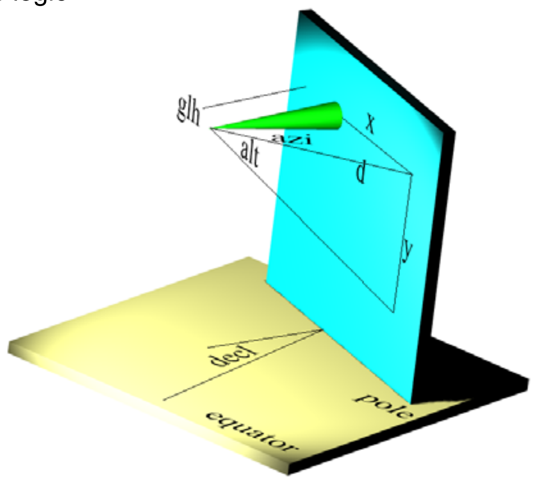
y = d * tan(altitude)

= tan(altitude) * glh / cos(azimuth)

```
' ######################################################################################
' #  MAIN LOOP   #   declination in this example is –23.44   (winter solstice)         #
' ######################################################################################

'# this uses NODUSX,Y from the gnomon drawing earlier
wx = 0                    '# these, when 0, tell the calendar
wy = 0                    '# line draw to draw nothing
hr = 7                    '# start at 0700 am (1st seg not drawn)

while hr < 17             '# end at 1700
   '# get an altitude and azimuth for this one hour at this SOLAR declination
   az =  azi( hr, latr, -23.44 )    ' degrees azimuth
   xxx=  glh * tan(2*3.1416*(az)/360)
   al =  alt( hr, latr, -23.44 )    ' degrees altitude
   yyy=  glh * tan(2*3.1416*al/360) / cos((2*3.1416*(az)/360))

   if wx<>0 and wy<>0 then
         xl=nodusx+wx
         yl=nodusy-wy
         xn=nodusx+xxx
         yn=nodusy-yyy
         ptb = CStr(xl)+","+CStr(yl)+","+CStr(0)
         pte = CStr(xn)+","+CStr(yn)+","+CStr(0)
         if abs(xl) < 130 and abs(xn)<130 then        ' all x values in range?
            if yl > -130 and yl<0 then                ' last Y in range ?
               if yn > -130 and yn<0 then             ' last Y in range ?
                  Set lineObj = ThisDrawing.ModelSpace.AddLine(ptb,pte)
               end if
            end if
         end if
   end if
   wx = xxx
   wy = yyy
   hr = hr + 0.25
wend
```

The process is similar to the horizontal dial process, and uses the base of the nodus and not dial center. The above code is written in Visual Basic Script as used in the free NanoCAD program. Some of the Illustrating Shadows CAD programs may use a different algorithm.

Vertical Declining dial ~ calendar or declination curve logic



As for the horizontal dial, the approach for a vertical declining dial is to select a solar declination and run the hours from morning to afternoon. Running the hours means using LAT (local apparent time), and for each time increment, derive the sun's azimuth and altitude. That part of the process is the same for the vertical, horizontal, and vertical declining dial. Assume "glh" is the gnomon linear height. For the vertical declining dial, the process is similar to the vertical dial, except the vertical dial plate, or wall, has a declination which is factored in. As before, the nodus base point is located at "nodusx,nodusy"

So: x = glh * tan(azimuth + wall declination)

And:- d = glh / cos(azimuth) which is needed for the "y" ordinate:-

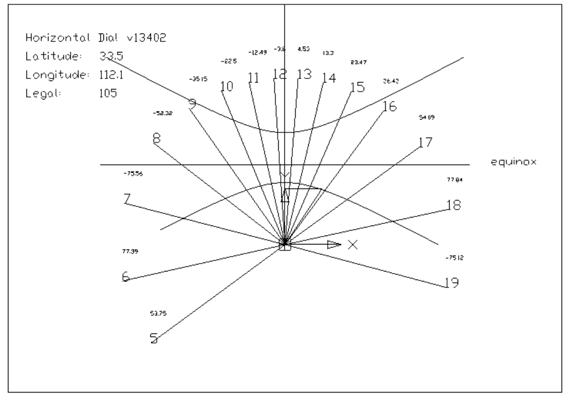
so: y = d * tan(altitude) = glh * tan(altitude) / cos(azimuth+wall declination)

```
#######################################################################################
#   MAIN LOOP   #   declination in this example is –23.44   (winter solstice)          #
#######################################################################################

# this uses NODUSX,Y from the gnomon drawing earlier
# set the summer and winter x,y coordinates to 0 to begin with
wx = 0                     # these, when 0, tell the calendar
wy = 0                     # line draw to draw nothing
hr = 7                     # start at 0700 am (1st seg not drawn)
# keep "decl" useful for azimuth on the dial plate
decl = dec
if dec >0:
        decl = -1 * dec
while hr < 19 :            # end at 1700 or 5pm
        # get an altitude and azimuth for this one hour at this winter SOLAR declination
        al =  alt( hr, lat, -23.44 )
        az =  azi( hr, lat, -23.44 )
        xxx =  glh * math.tan(math.radians(az+decl))
        yyy =  glh * math.tan(math.radians(al)) / math.cos(math.radians(az+decl))
        if dec > 0:
                xxx = -1 * xxx
        if abs(xxx)< 3 and abs(yyy)<2 :
                if wx<>0 and wy<>0:
                   c = Part.makeLine((nodusx+xxx, nodusy-yyy,0),(nodusx+wx, nodusy-wy, 0))
                   Part.show(c)
                wx = xxx
                wy = yyy
#
```

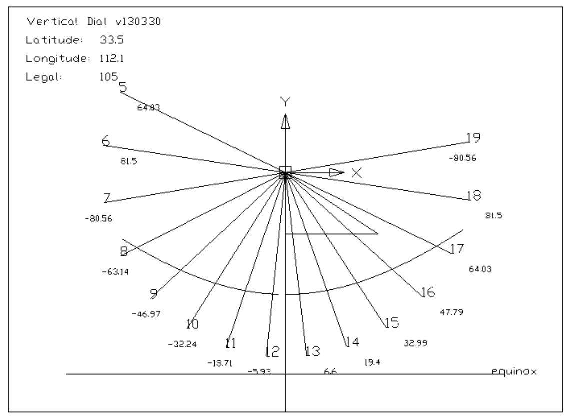
The process is similar to the vertical dial process, and uses the base of the nodus and not dial center. The above code is written in Python as used in the free FreeCAD program. Some of the Illustrating Shadows CAD programs may use a different algorithm. Python uses indentation rather than an approach using braces {…} as in Java Script, or some other code block terminator such as IF…END IF, or WHILE…WEND as used in Visual Basic.

Illustrating Shadows provides:- functions.js, functions.vbs, functions.py to aid programming.

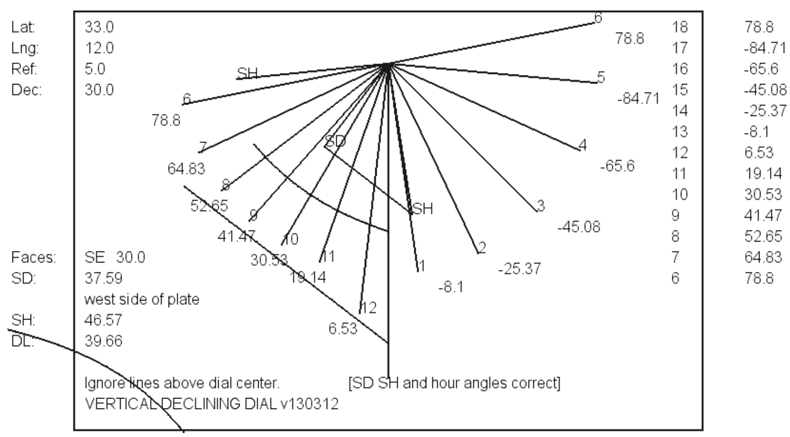
The Java Script produced horizontal dial with calendar curves created in NanoCAD:-



The Visual BASIC Script (vbs) produced vertical dial created in NanoCAD:-



The vertical declining dial using Python as in FreeCAD:-



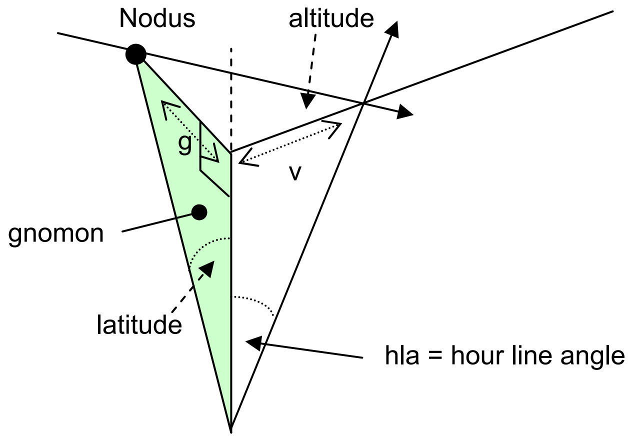
Both FreeCAD and NanoCAD are CAD programs available freely. FreeCAD is open source, and NanoCAD while free, does require a serial number (immediately available by email), and registration (immediately provided online). Both install easily, work first time, and do not need extra software to be downloaded. They have been tested on Windows 8 and earlier editions. Being free, documentation is somewhat limited, so Programming Shadows on the Illustrating Shadows web site or CD is almost essential for anyone programming in those systems.

Declination lines for the horizontal dial using trigonometry.

The declination line is nothing more than a depiction on the dial plate that shows the sun's declination. This varies by about plus or minus 23.5 degrees. It relates to the calendar date. Certain dates are significant, namely the winter and summer solstices, shortest day and shortest night respectively. The equinox has two dates and is when the day length is the same as the night time, however every day on the equator is an equinox. The solstice and equinox lines are helpful for other purposes, such as the Italian and Babylonian hours, discussed later in this chapter as well as in chapter 24. This section on trigonometric methods was the basis for appendix tables A4.4 and A4.5

<u>Using the SUN'S ALTITUDE and a unit gnomon linear height</u>

Given a gnomon linear height, how far from the nodus base (nodus dropped perpendicular to dial plate) on a horizontal or flat dial is the declination point?



tan (alt) = gnomon linear height / distance from gnomon base = g / v

v = g / tan(alt)

or in spreadsheet (Excel) terms

v = gnomonlinearheight / (TAN(ASIN(SIN(RADIANS(decl)) *SIN(RADIANS(lat))+COS(RADIANS(decl)) * COS(RADIANS(lat)) * COS(RADIANS(15*(12-hh)))))))

This method works for all angles, and assumes a known perpendicular distance from the nodus to the dial plate (gnomon linear height).

The tables for this method are in appendix A4.4 for latitudes 0 to 65 for declinations of +23.5, 0, and – 23.5

NOTE: The azimuth may be used as a vector from the nodus base, or using the law of sines, a distance from dial center along the hour line may be derived. However, the azimuth method fails when the azimuth is 0 or solar noon, it also fails at the equator. If using the dial center, then that dial center must be accessible, which is often not true for great decliners.

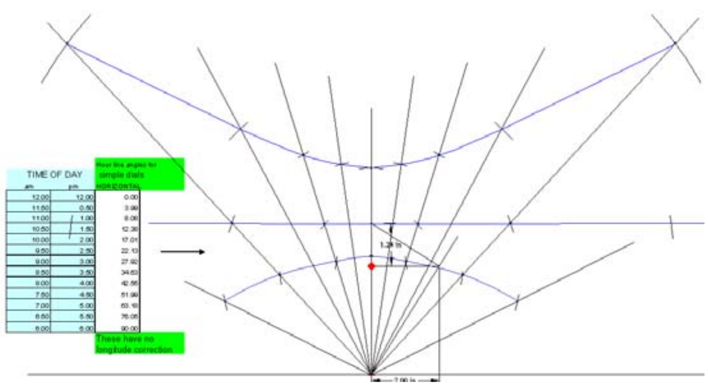
DEMONSTRATION OF THE ALTITUDE METHOD

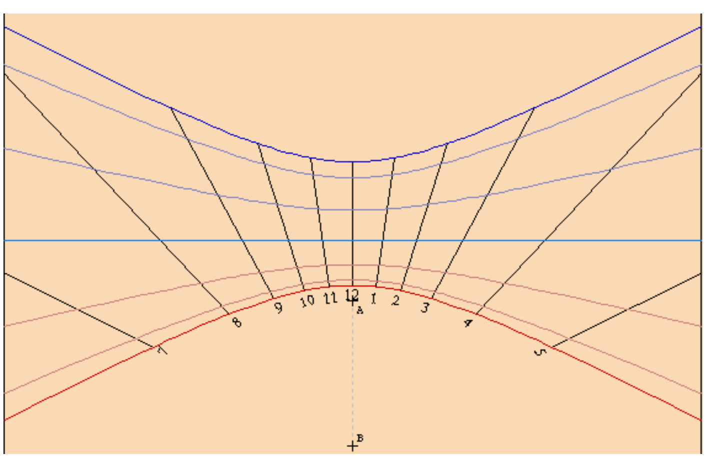
CONSIDER LATITUDE 32 – or a flat dial whose SH (style height) is 32 degrees

Distance from gnomon base to the hour line for the relevant declination point						chart 1
Gnomon linear height is:		**2**				
Latitude	Decl	23.5				
	1200	1100	1000	900	800	700
31	0.263	0.546	1.046	1.704	2.678	4.465
32	0.299	0.564	1.053	1.704	2.666	4.416
33	0.335	0.583	1.062	1.705	2.655	4.369

Latitude	Decl	0				
	1200	1100	1000	900	800	700
31	1.202	1.355	1.805	2.625	4.216	8.790
32	1.250	1.400	1.848	2.669	4.272	8.890
33	1.299	1.447	1.893	2.715	4.330	8.994

Latitude	Decl	-23.5				
	1200	1100	1000	900	800	700
31	2.804	3.006	3.701	5.345	10.468	
32	2.910	3.118	3.837	5.557	11.085	-199.623
33	3.022	3.236	3.981	5.787	11.780	



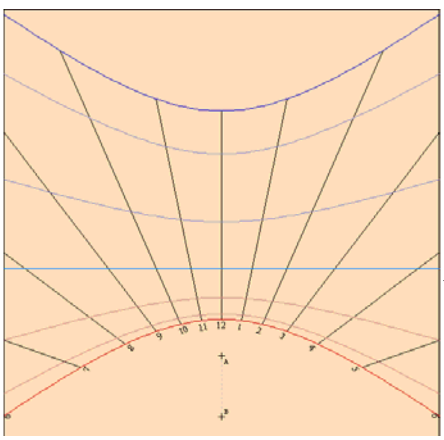


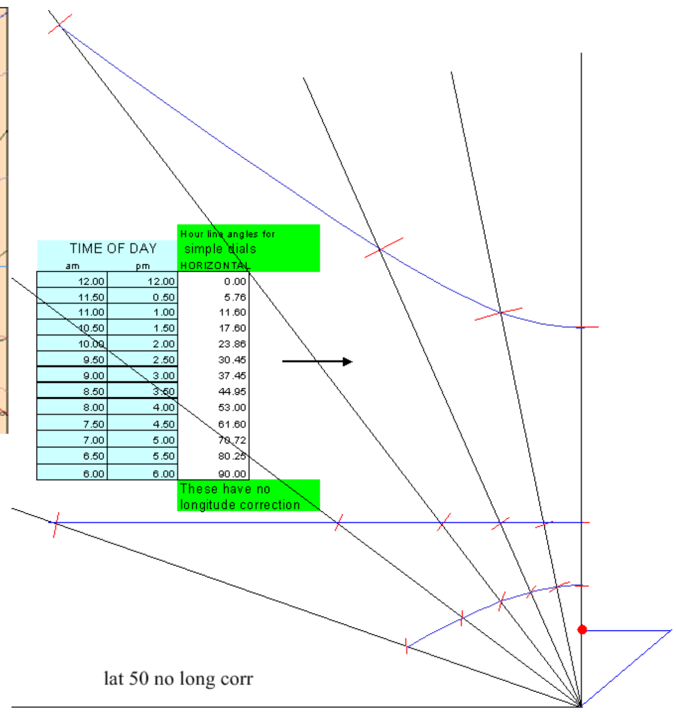
CONSIDER LATITUDE 50 – or a flat dial whose SH (style height) is 50 degrees

Distance from gnomon base to the hour line for the relevant declination point						chart 1
Gnomon linear height is:			**2**			
Latitude	Decl		23.5			
		1200	1100	1000	900	800	700
49		0.954	1.068	1.386	1.892	2.654	3.896
50		0.997	1.107	1.417	1.915	2.665	3.882
51		1.041	1.148	1.450	1.939	2.677	3.868

Latitude	Decl		0				
		1200	1100	1000	900	800	700
49		2.301	2.441	2.897	3.819	5.760	11.607
50		2.384	2.525	2.985	3.919	5.893	11.854
51		2.470	2.612	3.077	4.025	6.033	12.115

Latitude	Decl		-23.5				
		1200	1100	1000	900	800	700
49		6.343	6.852	8.864	15.941		
50		6.752	7.309	9.547	17.848	-186.520	-12.927
51		7.212	7.827	10.339	20.268		





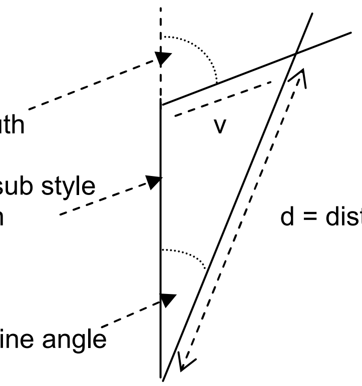
TIME OF DAY am	pm	HORIZONTAL
12.00	12.00	0.00
11.50	0.50	5.76
11.00	1.00	11.60
10.50	1.50	17.60
10.00	2.00	23.86
9.50	2.50	30.45
9.00	3.00	37.45
8.50	3.50	44.95
8.00	4.00	53.00
7.50	4.50	61.60
7.00	5.00	70.72
6.50	5.50	80.25
6.00	6.00	90.00

The appendices have tables A4.4 for latitudes 0 to 65 for declinations of 23.5, 0, and – 23.5

Declination lines for the horizontal using trigonometry (continued)

<u>Using the SUN'S ALTITUDE and AZIMUTH, and a unit GNOMON LINEAR HEIGHT</u>

Given a gnomon's linear height (nodus to dial plate), how far along an hour line from the dial center is the declination point on a horizontal dial (with certain standard declinations)

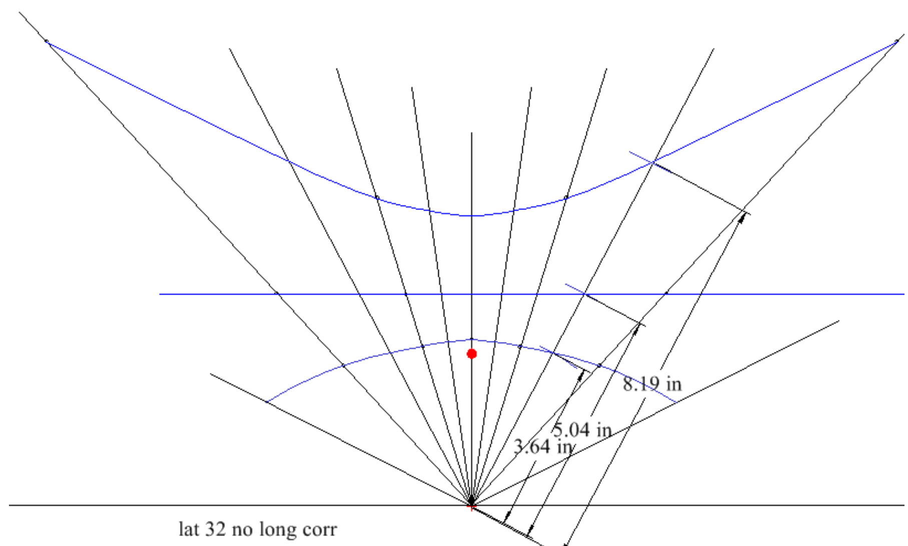


using the law of sines: d / (sin(180-azimuth) = v / (sin(hla))
d = v * sin(180-azimuth) / (sin(hla))

This method fails when the azimuth is 0 or solar noon, it does not work on the equator, and is only practical if the dial center is accessible. This is often not true for great decliners. For this reason, this method is not common.

The azimuth method can be used from the gnomon base as an angle which when extended intersects the appropriate hour line, or using the law of sines can be used to develop the distance along an hour line from dial center which is the declination point which is discussed here.

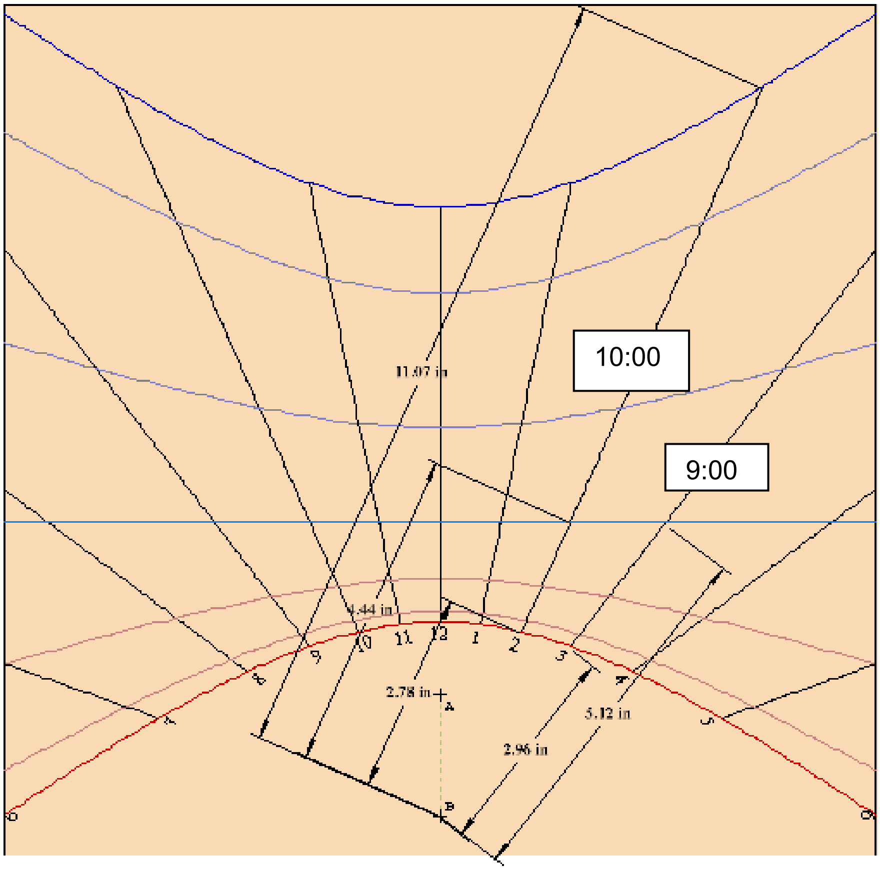
To the right is an example for latitude 32 using the distance from the dial center to the calendar point on the hour line. The gnomon linear height is **2** units.



Latitude	Decl	1200	1100	1000	0900	0800	0700
320	235		351	354	364	391	481
320	00		450	465	504	604	986
320	-235		625	678	818	1323	

chart 2

Another example is for latitude 50 and two hour lines have been selected to demonstrate that the measurements using the tables match dial generated with other methods. The gnomon linear height is **2** units.



lat 50 in SHADOWS

		12:00	11:00	10:00	9:00	8:00	7:00
50.0	23.5		2.70	2.78	2.95	3.31	4.10
50.0	0.0		4.15	4.44	5.12	6.75	12.30
50.0	-23.5		8.95	11.06	19.15		

10 and 2 9 and 3

chart 2

Declination

Note that the tables have the noon time blocked out, this is because that at noon (L.A.T.) the azimuth is zero.

The tables for this method are in appendix **A4.5** for latitudes 0 to 65 for declinations of 23.5, 0, and – 23.5

Declination lines for the horizontal using trigonometry (continued)

Using the sun's altitude and azimuth, and a unit style linear length

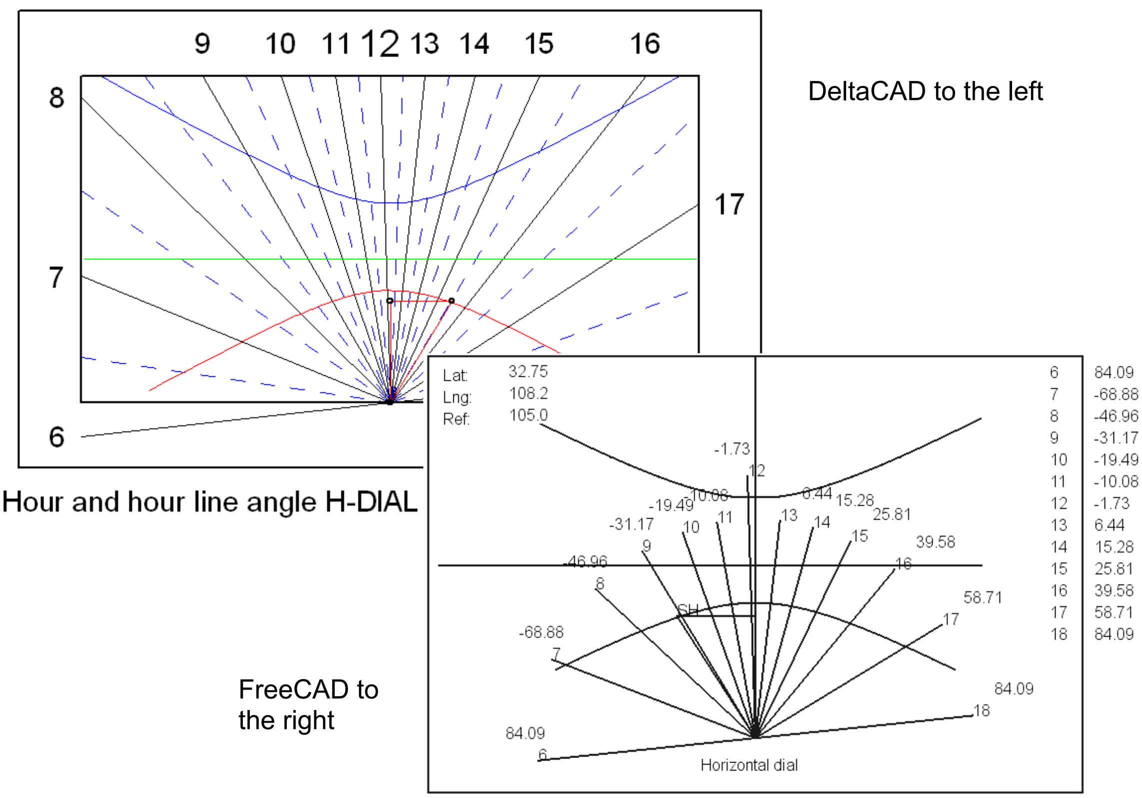
Given a style's linear length, how far along an hour line from the dial center is the declination point on a horizontal dial (with certain standard declinations)

This is a simple conversion from the style's length and latitude, to a gnomon height. The tables for this method are in appendix **A4.6** for latitudes 0 to 65 for declinations of +23.5, 0, and – 23.5

Procedural code compared to spreadsheet methods – horizontal dial

Using the sun's ALTITUDE AND AZIMUTH, and a unit style linear length ~ derived from the nodus to dial plate linear distance (gnomon linear height)

The formulae developed and used have been based on simple trigonometry and used as is or in spreadsheets. Such formulae can also be used in repetitive loops as in simple programming. While this book uses TurboCAD throughout, there is another CAD package called DeltaCAD which, while limited to 2d drafting, does provide an adaptation of the BASIC language integrated into their graphics system. Below is a picture captured using DeltaCAD, and the code, which is available on the CD accompanying this book as well as on this book's web site, has some points worth noting. Some of the key parts of the code are extracted and explained on the next page.



DeltaCAD to the left

Hour and hour line angle H-DIAL

FreeCAD to the right

Only an extract is shown from the DeltaCAD macro or program. The first point is that unlike some programs, this program shows hour lines bounded by a boundary box, and they are not constrained to the calendar lines or curves. Calendar lines use a nodus which can be the style tip but it does not have to be. A style can be longer, and thus provide more accuracy. For that reason this program treats hour lines independently from calendar data.

```
' -----------------------------------------------------------------
' PHASE 3: draw calendar lines
' -----------------------------------------------------------------
Dim wx, wy, qx, qy, sx, sy As single

' set the summer and winter x,y coordinates to 0 to begin with
wx = 0                           ' these, when 0, tell the calendar
wy = 0                           ' line draw to draw nothing
qx = 0                           ' if non zero then calendar lines
qy = 0                           ' are drawn
sx = 0                           ' and either way the three x,y pair
sy = 0                           ' are updated with this hours points

For hr = 6  To 17.9 Step 0.1

' =================================================================
    ' for the hour (hr) calculate the calendar data for this
    ' local apparent time (L.A.T.) which is NOT longitude adjusted for
    ' the reference longitude. This is ok because we are drawing
    ' calendar lines without regard to actual hour lines.

    ' This program draws legal time hour lines bounded by the box,
    ' and the calendar lines are based on the nodus and the calendar
    ' lines do not constrain the hour lines which thus allows a long
    ' style with a nodus partially long it.

    ' these formulae are meaningless at noon, and for winter lines they
    ' are meaningless at 6 am and pm local apparent time.

    If (hr < 11.5 Or hr > 12.5) Then
       ' this is a usable hour, we have three declinations (-23.5, 0, 23.5)
       ' we have the hour
       ' we have the gnomon linear height

       ' first get the winter, equinox, and summer distances on the L.A.T.
       ' hour lines, we do not draw the L.A.T. hour line.
       wz = Cal ( (hr), (lat), -23.5,  glh )  ' last parm = decl
       qz = Cal ( (hr), (lat),  0.0 ,  glh )  ' last parm = decl
       sz = Cal ( (hr), (lat), +23.5 , glh )  ' last parm = decl

       ' get the L.A.T. hour line angle also
       zh = Hla ( (hr), (lat), (0), (0) )     ' long=ref=0 means L.A.T. hours
       ' and zh can be negative (am) or positive (pm)

       ' we have an hour line angle (L.A.T.), and a distance
       '
       '  nwx nqx nsx
       '
       '    wz \      |       nwy
       '        \     |
       '     qz  \ zh |       nqy
       '          \   |
       '       sz  \  |       nsy
       '            \ |
       '             *

       ' assuming a center of 0,0 (dial center) we can calculate the
       ' x,y values for each declination by...
       '
       ' x = sin(h) * z
       ' y = cos(h) * z

       nwx = Sin(rad((zh))) * wz
       nqx = Sin(rad((zh))) * qz
       nsx = Sin(rad((zh))) * sz

       nwy = Cos(rad((zh))) * wz
       nqy = Cos(rad((zh))) * qz
       nsy = Cos(rad((zh))) * sz
```

Calendar curves can get unmanageable, and that brings to mind the best way of drawing them. In order of best practices, the techniques used might be:-

- Draw each segment and whenever an "x" or "y" ordinate exceeds the boundaries, then derive a new x,y pair that starts at the boundary.
- The method here is to not draw any line segment if any of the two x,y coordinates exceed a boundary
- Use limiting hours

```
        If (wx<>0 And wy<>0 And qx<>0 And qy<>0 And sx<>0 And sy<>0) Then
           ' this is not the first time around so we can draw a line
           ' from (sx,sy) to (nsx,nsy) and so on

           ' the calendar lines can get excessive
           ' the best code would take each line segment and if it impacts
           ' a boundary then shorten the line segment.
           ' the next best code is probably what is in Function Dln
           ' Then the next best would be a table of latitudes and what
           ' hours are acceptable as limits, and so on.

           ' This code doesn't like winter lines at higher latitudes,
           ' so it tests the "y" values against the equinox Y value

           dcsetlineparms dcblue,dcsolid,dcthin   ' page 228 Manual
           If (wy > qy) And (nwy > qy) Then
              ok = Dln ( (wx), (wy), (nwx), (nwy) )
           End If

           ' eqxinox lines can also get excessive
           dcsetlineparms dcgreen,dcsolid,dcthin  ' page 228 Manual
           ok = Dln ( (qx), (qy), (nqx), (nqy) )

           dcsetlineparms dcred,dcsolid,dcthin    ' page 228 Manual
           ok = Dln ( (sx), (sy), (nsx), (nsy) )

        End If

        ' and make these new points be the start for the next calendar line's points
        wx = nwx
        wy = nwy
        qx = nqx
        qy = nqy
        sx = nsx
        sy = nsy

     End If
  ' =========================================================================
  Next hr
```

The above extract from the code is subject to criticism. That code was written to illustrate teaching points. While structured code was used throughout (no GOTO statements), and indented logical levels were used for the most part, there are violations of good code practice. For example it is poor practice to code hard numbers inline in the code, such constants should be declared in a data area, and assigned values similarly. Examples of that violation are the hour range in the FOR ... NEXT process where the starting, ending, and step increase are hard coded.

Also, tricks were used to simplify data type mismatches, which is why some values were in (parentheses) in some function calls. There are better practices than that. Obviously, some functions used above are not shown, they are included in the complete macro or program on the CD accompanying this book, as well as on this book's web site.

The objective of the above code is simply to show formulae being used in iterative or repetitive code, and to show poor as well as good programming techniques.

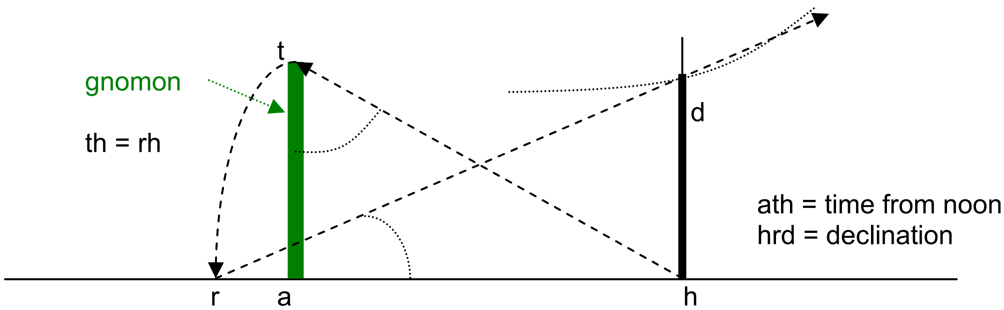
The FreeCAD program has similar code which is worth studying, FreeCAD uses Python.

ITALIAN LINE CONSTRUCTION

The Italian hours were the hours from the previous sunset. However, practical usage has them as being the hours until the next sunset. Yours is the choice. The method of drafting them is in essence to calculate the LAT (local apparent time) of sunset for the solstices, and backing off one hour at a time, calculate the point on the solstice curves for a line, and draw it. In practice, it may be that the points might be one solstice point, and one equinox point, or even something else, this is because for some dials, the lines may extend to infinity. For a human, this is simple. For a computer, there is a bit more to it. The mere act of locating where an hour line intersects a solstice (or other curve), is a bit more involved.

NOTE: The programming code that follows are simplified extracts, and do not include various cross checks as regards coordinates exceeding the drawing area. The "sqr" function is square root.

Polar Dial ~ the process is relatively simple. The geometric method for constructing calendar or declination lines is as depicted below, along with a trigonometric formulae development:-



we see that:- tan (hrd) = dh / rh = dh / th, thus dh = th * tan (declination)

and:- cos (ath) = ta / th thus th = ta / cos (ath)

so given: dh = th * tan (declination)
then:- dh = ta * tan (declination) / cos (time)

in other words the distance up the hour line for the point on which the declination (calendar) line will lie is equal to the style height times the tangent of the declination all divided by the cosine of the time.

```
' ------------------------------------------------------------------------
' locating Italian line points - wx and wy – winter ordinates – decl = -23.44
' ------------------------------------------------------------------------

' ss = sunset time from noon
' hr = an hour LAT (being n*hrs from sunset for different Italian lines)

' *** calculate hour angle of sunrise/set ~ ~ decl = -23.44 winter solstice
decl =  -23.44
ss   = (deg((acs( Tan(rad(lat)) * Tan(rad(-decl))  ))) / 15)
hr   = ss – hrFromSS

' get an X and Y for the hour's intersection with the declination curve
nwx  = glh * Tan(rad(15*(hr)))                     ' "h" in the above
nwy  = (-1)* Tan(rad(decl))*Sqr((glh*glh)+(nwx*nwx)) ' "d" in the above
' alternative formula from the one shown above
```

Meridian dial ~ the process is similar to the polar dial except that for a polar dial noon is at the center of the dial plate where the solstice curves are closest, whereas for a meridian dial, 6am and 6pm are where the solstice curves are closest.

Horizontal Dial ~ the process is a bit more involved. The hour lines radiate at angles from the dial center, so the means of locating the point where the hour line intersects the calendar or declination curve need a bit more effort compared to the polar dial. In essence, the altitude and azimuth of the sun are found for the time and solar declination for the specific latitude. From this information, the x and y ordinates are found.

```
' ------------------------------------------------------------------------
' locate Italian line points - wx and qx (and y) are two end points
' ------------------------------------------------------------------------

' hr = an hour LAT (being n*hrs from sunset for different Italian lines)
' having an hour line angle (LAT), and a distance

' *** calculate hour angle of sunrise/set ~ ~ declination= -23.44 winter solstice
decl =  -23.44
hr = 24- (deg((acs( Tan(rad(lat)) * Tan(rad(decl))  ))) / 15)
hr = hr – hrFromSS

‘ **( calculate a distance along an hour line
wz = Cal ( (hr), (lat),  decl  ,  glh )  ' dist from dial center for cal point
‘    Cal ~ determines altitude and azimuth

wh = Hla ( (hr), (lat),  (0)    , (0) )  ' long=ref=0 means LAT hours
nwx = Sin(rad((wh))) * wz
nwy = Cos(rad((wh))) * wz
```

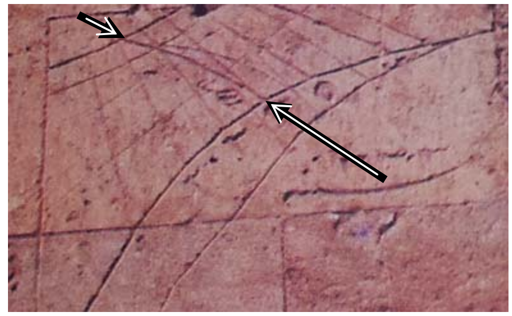
Vertical Dial ~ the vertical dial is a bit more involved than the horizontal, because azimuth and altitude has to be projected onto a vertical surface, not a simple horizontal plate.

```
' having an hour line angle (LAT), and a distance
'
'    nwx,nwy-----+
'        \       |
'      wz  \   wh|
'            \   |
'              \ |
'                *
'
' a dial center of 0,0 then calculate the x,y values for each declination as...
'
' x = sin(h) * z
' y = cos(h) * z


' *** calculate hour angle of sunrise/set ~ declination= -23.44 winter solstice
decl = -23.44
' hr = an hour LAT (being n*hrs from sunset for different Italian lines)
hr = -(deg((acs( Tan(rad(90-lat)) * Tan(rad(decl))  ))) / 15 )  + hrFromSS

wz = Cal ( (hr), (lat),  decl  ,  glh )    ‘ uses altitude and azimuth
wh = Hla ( (hr), (lat),  (0)    , (0) )
nwx = Sin(rad((wh))) * wz
nwy = Cos(rad((wh))) * wz
```

Vertical Declining Dial ~ the vertical decliner is more involved than the vertical dial, because azimuth and altitude has to be projected onto a vertical surface that is declined.

For specific details, the DeltaCAD macros should be studied.

ISLAMIC PRAYER TIME DIALS



Just as the Roman Catholic Church used mass dials to determine the time of mass, and in some cases, other offices as well, so also did the Islamic world, a world away, use them for their prayer times. Have you ever wondered what a curve on a middle eastern sundial might indicate? The answer lies in the Muslim tradition of marking afternoon prayer, asr or عصر, by sun shadow indications. Picture courtesy of Osmanli Gunes Saatleri.

The following prayer time definitions are often used.

Times of Prayers

فجر	شروق	ظهر	عصر	مغرب	عشاء
FAJR	SHURUQ	ZUHR	ASR	MGRIB	ISHA
Dawn Prayer	Sunrise	Noon Prayer	After-noon Prayer	Sunset Prayer	Night Prayer
AM	AM	PM	PM	PM	PM

Since the sun is needed for a sundial to work, the dawn prayer FAJR would not be on a sundial, similarly, the night prayer ISHA would not be marked either.

So that leaves:-

شروق	ظهر	عصر	مغرب
SHURUQ	ZUHR	ASR	MGRIB
Sunrise	Noon Prayer	After-noon Prayer	Sunset Prayer
AM	PM	PM	PM

Sunrise can be marked but it would be somewhat intuitive, since actual sunrise is easy to observe, similarly with sunset. Sunrise and sunset can be marked on horizontal, equatorial, and polar dials. A vertical decliner can mark either the sunrise or the sunset, but not both unless two dial plates were used. A vertical dial facing the equator would not show sunrise and sunset except on the fall and winter sides of the equinoxes. So that leaves:-

ظهر	عصر
ZUHR	ASR
Noon Prayer	After-noon Prayer
PM	PM

And here come two prayer times that can be marked. Noon is somewhat simple, and as it happens, at least in the context of sundials, is the basis for the afternoon prayer.

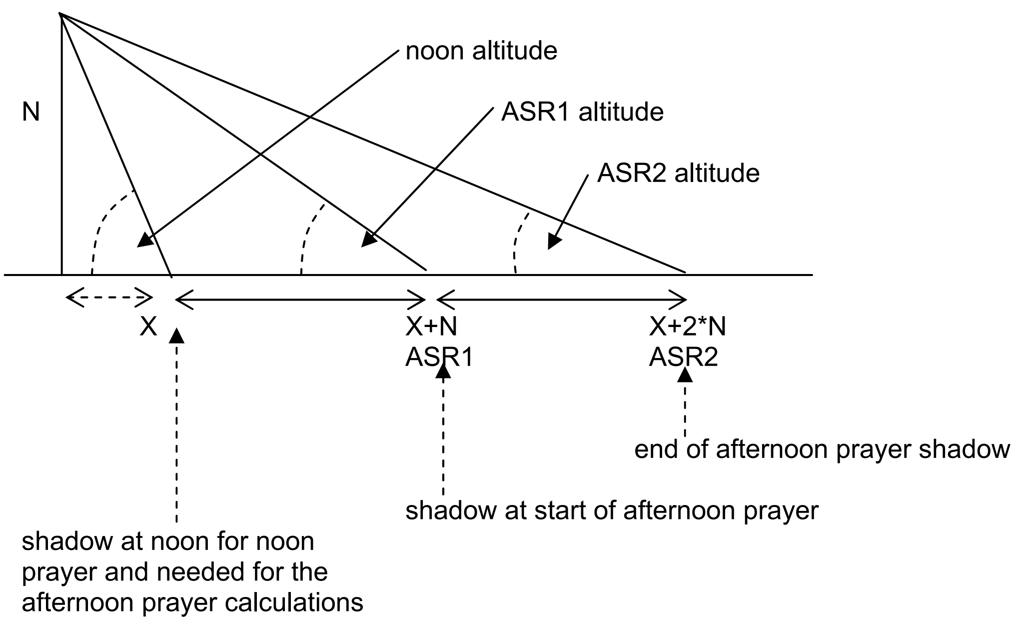
Noon Prayer	ZUHR	ظهر

Noon is easily portrayed on almost any sundial, the exception being the meridian dials. So that problem is solved, except some information from it is needed for defining the period of Afternoon Prayer. And it is the afternoon prayer that presents a problem for the diallist.

Afternoon Prayer	ASR	عصر

Afternoon Prayer, ASR, was defined as starting at a point where the shadow of a vertical rod of length “N” is equal to the same rod’s shadow at noon, plus the length “N” of the rod. And it ends at a time when the shadow of that rod is equal to the noon shadow length plus twice the length of the rod.

Pictorially the angles would look somewhat like the following, however, it should be remembered that the ASR1 and ASR2 points would be at different solar azimuths.



So far this has been non latitude specific.

		shadow distance.....
point X	noon prayer	= X
point ASR1	afternoon prayer start	= X + N
point ASR2	afternoon prayer end	= X + N + N

The altitudes, needed to compute “X” and thus ASR1 and ASR2 would then be:-

		sun’s altitude
point X	noon prayer	= noon solar altitude
point ASR1	afternoon prayer start	= atan(N / (X+N))
point ASR2	afternoon prayer end	= atan(N / (X+2*N))

Afternoon Prayer, ASR, defined as starting at a point where the shadow of a vertical rod of length “N” is equal to the same rod’s shadow at noon, plus the length “N” of the vertical rod.

And it ends at a time when the shadow of that rod is equal to the noon shadow length plus twice the length of the rod.

The table to the right is an extract of a non latitude specific set of data, for every 5 degrees of solar altitude at noon. It is extracted from the spreadsheet:-

XLS bk3 sup Islamic prayer times.xls

Again, the ASR1 and ASR2 distances are at different azimuths, which is what makes things interesting.

Non latitude specific ASR prayer time data

rod length N:	1	noon shadow = N/TAN(altitude)	
Noon solar altitude	shadow length on the ground X	Shadow length at ASR1, start of afternoon prayer	Shadow length at ASR2, end of afternoon prayer
altitude	X	=X+N	=X+2*N
90	0.000	1.000	2.000
85	0.087	1.087	2.087
80	0.176	1.176	2.176
75	0.268	1.268	2.268
70	0.364	1.364	2.364
65	0.466	1.466	2.466
60	0.577	1.577	2.577
55	0.700	1.700	2.700
50	0.839	1.839	2.839
45	1.000	2.000	3.000
40	1.192	2.192	3.192
35	1.428	2.428	3.428
30	1.732	2.732	3.732
25	2.145	3.145	4.145
20	2.747	3.747	4.747
15	3.732	4.732	5.732
10	5.671	6.671	7.671
5	11.430	12.430	13.430
0			

So, now we need the place specific altitudes. The formula for altitude is:-

alt = ASIN(SIN(dec) * SIN(lat) + COS(dec) * COS(lat) * COS(lha))

and at local apparent noon (no longitude correction, no mean time correction) the local solar hour angle (lha) is 0 which makes cos(lha) = 1 so, the formula for the noon altitude then is:-

noonAltitude =ASIN(SIN(dec)*SIN(lat)+COS(dec)*COS(lat))

So now the table becomes:-

		sun’s altitude
afternoon prayer start	ASR1	= atan(N / (X+N))
afternoon prayer end	ASR2	= atan(N / (X+2*N))
since... noon altitude		= ASIN(SIN(dec)*SIN(lat)+COS(dec)*COS(lat))
and since..		tan(noonAltitude) = N / X
then	X	= N / tan(noonAltitude)

which has at this point become latitude specific as well as solar declination specific because the sun’s altitude at noon depends on both latitude and solar declination.

The next stage is how can this be depicted on a sundial plate, and a horizontal dial is a good place to start.

From the preceding page, the tip of the shadow rod will fall at an altitude:-

noonAltitude = ASIN(SIN(dec)*SIN(lat)+COS(dec)*COS(lat))

and the key shadow's distance at noon "X" is

X = N / tan(noonAltitude)

= N / tan(ASIN(SIN(dec)*SIN(lat)+COS(dec)*COS(lat)))

which makes the other altitudes:-

ASR1 = atan(N / (X+N)) start afternoon prayer

and

ASR2 = atan(N / (X+2*N)) end afternoon prayer

So for a given latitude, and declination, the results for ASR1 would look like:-

	B	C	D	E	F	G	H	I
2								
3	latitude	38.9		JD	21			
4				dec	-20.0859			
5								
6								
7	correction to standard mean time mm.mm				0		rod height	1
8							"N"	
9								
10	noon altitude		31.01408		noon shadow "X"		1.66335	
11								
12	thus the ASR1 shadow distance is				shadow X+N		2.66335	
13								
14	thus the ASR1 shadow's altitude is				20.57948			

Altitude and distance are all related. To get a coordinate of the point for the date (i.e. solar declination), an azimuth is needed. But the azimuth needs the local hour angle, so, we need the local hour angle for each altitude.

Since altitude = ASIN(SIN(dec) * SIN(lat) + COS(dec) * COS(lat) * COS(lha))

this can be solved for local hour angle:-

altitude = asin(sin(dec) * sin(lat) + cos(dec) * cos(lat) * cos(lha))

sin(alt) = sin(dec) * sin(lat) + cos(dec) * cos(lat) * cos(lha)

sin(alt) – sin(dec) * sin(lat) = cos(dec) * cos(lat) * cos(lha)

so

$$\frac{\sin(alt) - \sin(dec) * \sin(lat)}{\cos(dec) * \cos(lat)} = \cos(lha)$$

thus

lha = acos ((sin(alt) – sin(dec) * sin(lat)) / (cos(dec) * cos(lat)))

So the table on the preceding page expands to:-

	B	C	D	E	F	G	H	I
2								
3	latitude	38.9		JD	21			
4				dec	-20.0859			
5								
6								
7	correction to standard mean time mm.mm				0		rod height	1
8							"N"	
9								
10	noon altitude		31.01408		noon shadow "X"		1.663353	
11								
12	thus the ASR1 shadow distance is				shadow X+N		2.663353	
13								
14	thus the ASR1 shadow's altitude is				20.57948			
15								
16	then the local solar hour angle is				39.10654			

Since there is a solar hour angle, and each 15 degrees of that makes one hour, then the local apparent time from noon is the local hour angle times 15. Then, standard time which corrects for longitude is next, even though not used much these days, and finally the equation of time (EOT) is added in providing the standard mean time.

The latitude and longitudes used are for the Islamic Center in Washington, DC, and the following spreadsheet provides the final values for ASR1.

latitude	38.9	JD	21	longitude	77.05	
		dec	-20.085916	legal	75	
		eot	10.929331	dif mm	8.2	
correction to standard mean time mm.mm			19.129331	rod height	0.2	
				"N"		
noon prayer time	12.19	ظهر	ZUHR			
noon altitude	31.014084		noon shadow "X"		0.3326706	
	nAlt				nShX	
ASR	thus the ASR1 shadow distance is		shadow X+N		0.5326706	aShDist
thus the ASR1 shadow's altitude is		aAlt	20.579477	and azimuth	39.255609	aAzi
then the local solar hour angle is		aAlha	39.106544			
for afternon prayer ASR	aAhr	HRS pm	2.607	decimal	Local Apparent Time	
		HRS hh.mm	2.36	from noon	14.36	hh.mm
longitude adjusted sundial time		hrs+ long	2.744	decimal	Standard Time (sundial)	
		hh.mm	2.45	from noon	14.45	hh.mm
clock time		plus EOT	2.926	decimal	Standard Mean Time	
		hh.mm	2.56	from noon	14.56 ASR1 عصر	

And when compared to the Washington DC Islamic Center's web site the times for the noon and the ASR services agree within one minute, thus the formula and process are valid. This process is repeated for ASR2 where the altitude is lower and the distance to the shadow tip longer.

Times of Prayers

			فجر	شروق	ظهر	عصر	مغرب	عشاء
			FAJR	SHURUQ	ZUHR	ASR	MGRIB	ISHA
Days	ربيع الاول AH 1434	JAN 13 -27, 2013	Dawn Prayer	Sunrise	Noon Prayer	After-noon Prayer	Sunset Prayer	Night Prayer
			AM	AM	PM	PM	PM	PM
			hr. min	hr. min	hr.min	hr. min	hr. min	hr. min
Mon	9	21	5:42	7:22	**12:19**	**2:57**	5:17	6:48

The noon time on the web site it simply noon plus the longitude correction of 8.2 minutes and the EOT for the day of 10.9 minutes, totaling 19.1 minutes after noon.

At this point then, we have deduced the distances, and the hour angle hence the hour line angle, for ASR1, and ASR2 if desired follows the same process. The tip of the shadow rod will fall at an altitude:-

noonAltitude = ASIN(SIN(dec)*SIN(lat)+COS(dec)*COS(lat))

and the key distance "X" of the noon shadow is:-

X = N / tan(noonAltitude)
= N / tan(ASIN(SIN(dec)*SIN(lat)+COS(dec)*COS(lat)))

ASR1 altitudes are:-

ASR1 = atan(N / (X+N))

and the local hour angle around the north south polar axis is:-

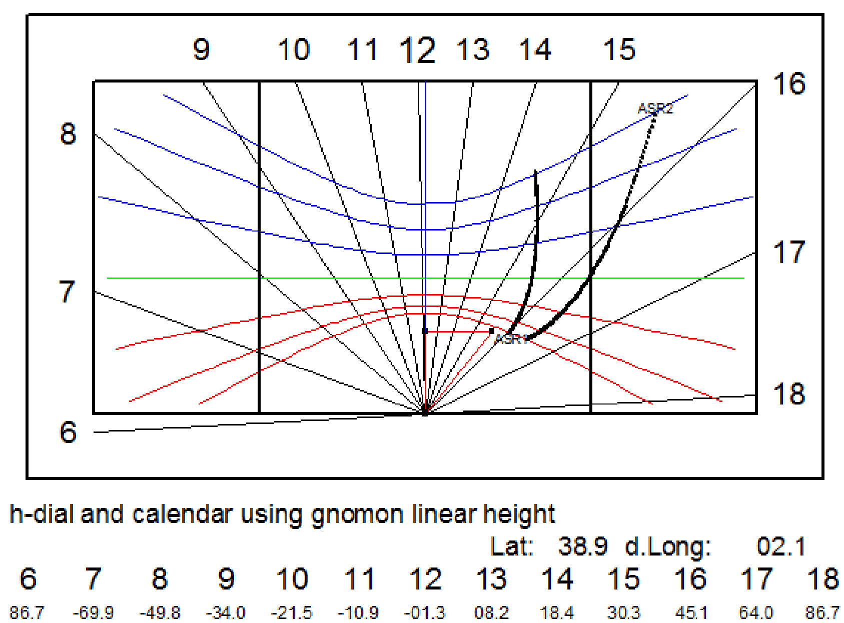
lha = acos ((sin(ASR1) – sin(dec) * sin(lat)) / (cos(dec) * cos(lat)))

This means that since we have a local hour angle, we then have the local hour line angle for the dial plate, which for a horizontal dial plate is:

hour line angle = atan (sin(lat) * tan(15*lha)

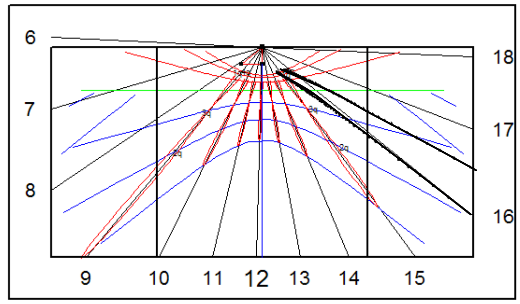
Since we now have a local hour line angle, and a distance along that hour line, then we have a coordinate for that point which can be plotted on a dial plate.

At this point, a DeltaCAD macro might prove useful to show how ASR1 looks during the year for declinations between -23.44 to +23.44, namely the solstices. Especially as the dial plate could be used as is. And similarly, the end of afternoon prayer, ASR2, is also depicted.



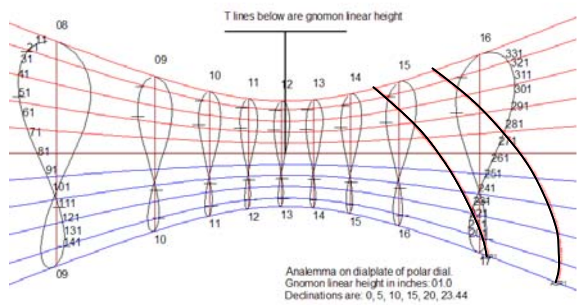
The DeltaCAD macro being:- BAS hDialFurniture [hDial choice 3].bas
And the Excel spreadsheet is:- XLS bk3 sup Islamic prayer times.xls

Of course, the same principles apply for a vertical dial, however the coordinates for the ASR1 points are not the same because for the horizontal dial azimuth results in a radial, for a vertical dial, it results in a horizontal displacement.



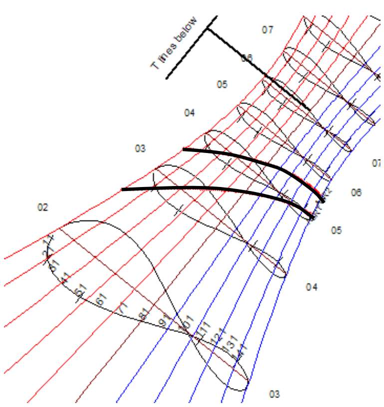
The logic works things out using the horizontal shadow length, and thus asr altitudes, and azimuth data. However, once calculated, the logic converts the azimuth not to a radial, but to a distance along the X axis, and then converts the altitude to a vertical distance which is dependent not only on the altitude, but also on the X value for the azimuth. The result is shown below. The vertical dial plate is shown above, it uses the DeltaCAD macro:- BAS vDialFurniture [vDial choice 7].bas

It may have been observed that this article has reduced its discussion of ASR2, mainly because it is the same process, and also that the calendars in current use often do not display it.



Of course, the afternoon prayer start and end times can be displayed on equatorial dials, and other such configurations, each has its opportunities for frustration. The polar depiction is shown to the right, and the logic for horizontal, vertical, and polar dials is included at the end of this article. The polar dial uses the DeltaCAD macro:-
BAS pDialFurniture [pDial choice 2].bas

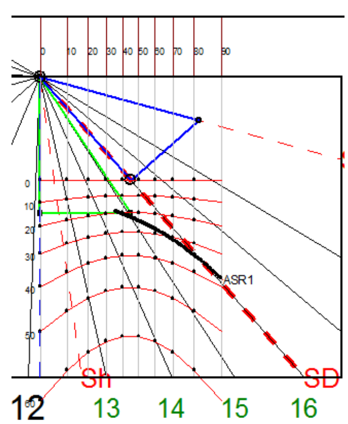
The meridian dial depicted below to the left, uses the DeltaCAD macro:-
BAS pDialFurniture [pDial choice 2].bas



To the right is the vertical decliner ASR depiction, the macro being:-

BAS vDecFurniture [vDial choice 9].bas

See Programming Shadows for some additional notes and logic.



The ASR times are latitude dependent. Thus, like the Italian and Babylonian hour lines, they are not valid when the dial is moved to another latitude. So, a horizontal dial with ASR lines (curves) designed for say latitude 33, cannot be moved to latitude 43, and tilted by ten degrees. While the hour lines would be thusly corrected, the Italian, Babylonian, and ASR lines and curves would no longer be correct.

Reference Material

Sundials, History, Art, People, Science   Mark Lennox-Boyd 2005
Osmanli Gunes Saatleri, Doc. Dr Nuseret CAM

http://theislamiccenter.com/worship/prayer-times/	times DC
http://iccpaz.com/prayer-times/	times PHX
http://www.akat.org/ast_tarihinden/osmanli_zamani/gunes_saati/	figures
http://mekaniksaat.wordpress.com/category/osmanlida-gunes-saati/	pictures

Medieval Roman Catholic service times and similar Islamic prayer times

Sext	at noon	Zuhr	ظهر
Nones	at 3pm	Asr	عصر
Vespers	(sunset evening service)	Magrhib	مغرب

Useful formulae at this point are:-

ALTITUDE: alt = ASIN(SIN(dec) * SIN(lat) + COS(dec) * COS(lat) * COS(lha))

AZIMUTH: azi = ATAN(SIN(lha) / ((SIN(lat) * COS(lha)) – (COS(lat) * TAN(dec)))

Formulae and cell references in spreadsheet:-   XLS bk3 sup Islamic prayer times.xls

cell C3	latitude
cell F3	Julian day
cell I3	longitude
cell I4	legal meridian's longitude
cell I7	rod height, gnomon linear height

dec	=DEGREES(0.006918-0.399912*COS(2*3.1416*(F3-1)/365)+0.070257*SIN(2*3.1416*(F3-1)/365)-0.006758*COS(2*2*3.1416*(F3-1)/365)+0.000907*SIN(2*2*3.1416*(F3-1)/365)-0.002697*COS(3*2*3.1416*(F3-1)/365)+0.00148*SIN(3*2*3.1416*(F3-1)/365))	
eot	=-1*(9.84*SIN(RADIANS(2*(360*(F3-81)/365)))-7.53*COS(RADIANS(360*(F3-81)/365))-1.5*SIN(RADIANS(360*(F3-81)/365)))-0.3	
nAlt ~ noon altitude	=DEGREES(ASIN(SIN(RADIANS(F4))*SIN(RADIANS(C3))+COS(RADIANS(F4))*COS(RADIANS(C3))))	cell D10
nShX ~ noon shadow "X"	=I7/TAN(RADIANS(D10))	cell H10
aShDist ~ ASR1 shadow distance	=H10+I7	cell H12
aAlt ~ ASR1 shadow alt	=DEGREES(ATAN(I7/(H12)))	cell F14
aAlha ~ asr1 local hour angle	=DEGREES(ACOS((SIN(RADIANS(F14))-SIN(RADIANS(F4))*SIN(RADIANS(C3))) / (COS(RADIANS(F4))*COS(RADIANS(C3)))))	cell F16
aAzi ~ ASR1 azimuth	=DEGREES(ATAN(SIN(RADIANS(15*F27))/(SIN(RADIANS(C3))*COS(RADIANS(15*F27))-TAN(RADIANS(F4))*COS(RADIANS(C3)))))	cell H14
aAhr ~ asr1 local hour	=F16/15	cell F17

Formulae and code in the Horizontal Dial DeltaCAD macro:-

BAS hDialFurniture [hDial choice 3].bas

The logic is that the ASR altitude and azimuth are used to generate a point which is then marked on the dial plate.

```
' code for Islamic asr1 Jan 21 2013
If iwantasr <>0 Then
   ' ---------------------------
   ' create a point at the asr1 radial(azimuth) and distance
   ' ---------------------------
   Dim nAlt     As Single     ' looks good
   Dim nShX     As Single
   Dim aAlt     As Single
   Dim aAlha    As Single
   Dim aAhr     As Single
   Dim aAzi     As Single
   Dim aShDist  As Single

   For zzz = -23.44 to 23.44 step 0.5
   '==========================
       '  zzz  is the declination here
       nAlt      = DEG(ASN(Sin(RAD(zzz))*Sin(RAD(lat))+Cos(RAD(zzz))*Cos(RAD(lat))))
       nShX      = glh/Tan(rad(nAlt))
       aAlt      = DEG(Atn(glh/(nShX  + glh)))
                   ‘ but use this for “asr2”
                   ‘          aAlt      = DEG(Atn(glh/(nShX + nShX + glh)))

       aAlha     = DEG(ACS(  (Sin(RAD(aAlt))-Sin(RAD(zzz))*Sin(RAD(lat)))  /  ( Cos(RAD(zzz))*Cos(RAD(lat))  )  ))
       aAhr      = aAlha/15                ' do this even though undone in aAzi to match .xls
       aAzi      = DEG(Atn(Sin(RAD(15*aAhr))/(Sin(RAD(lat))*Cos(RAD(15*aAhr))-Tan(RAD(zzz))*Cos(RAD(lat)))))
       '                aAzi is the radial from the base of the gnomon (dial center + glh/tan(rad(lat))
       '                aAzi is one ordinate for the asr1 point on the dial plate (i.e. direction
       aShDist   = glh + nShX
       'aAzi & asr1ShadowDist are coordinated from the base of the nodus to the azr1 point for this decl & latitude
       '                base of the nodus = glh / tan(radans(lat))
       xxx = aShDist * Sin(rad(aAzi))
       yyy = aShDist * Cos(rad(aAzi))
       If aAzi   < 0 Then
          aAzi = - aAzi
          xxx  = aShDist  * Sin(rad(aAzi))
          yyy  = -aShDist * Cos(rad(aAzi))
       End If

       dcCreateCircle  0+xxx , yyy+nodusy,  0.003

   '==========================
   Next zzz

   dcSetTextParms  dcBLACK,"Ariel","Bold",0,4, 21,0,0
   dcCreateText    0+xxx , yyy+nodusy-0.02,  0,  "ASR1"
   dcSetLineParms  dcRED,dcSOLID,dcTHIN

' end code for Islamic asr1 Jan 21 2013
End If
```

The Christian perspective

- "CHAPTER XVI,How the Work of God Is to Be Performed during the Day
- As the Prophet saith: "Seven times a day I have given praise to Thee" (Psalms 119:164), this sacred sevenfold number will be fulfilled by us in this wise if we perform the duties of our service at the time of Lauds, Prime, Tierce, Sext, None, Vespers, and Complin;"
- These times were based on the position of the sun and shadow lengths: dawn, sunrise, mid-morning, noon, mid-afternoon, sunset and nightfall. Moslem prayer times used the same time system but generally dropped the sunrise and mid-morning prayers.

The Islamic perspective

- http://www.al-islam.org/encyclopedia/chapter7/6.html

Formulae and code in the Vertical Dial DeltaCAD macro:-

BAS vDialFurniture [vDial choice 7].bas

The logic is the same as for the horizontal dial until the dial plate work is required, and at that point the final asr1 or asr2 altitude is converted to an X value on the dial plate and the Y value needs that X value as well as the altitude. Also, the program sets the latitude to the co-latitude because it used horizontal logic for the vertical dial, and that is why this code uses 90 – latitude in place of the latitude.

```
' code for Islamic asr1 Jan 24 2013
If iwantasr <> 0 Then
   ' --------------------------
   ' create a point at the asr1 radial(azimuth) and distance
   ' --------------------------
   Dim nAlt     As Single     ' looks good
   Dim nShX     As Single
   Dim aAlt     As Single
   Dim aAlha    As Single
   Dim aAhr     As Single
   Dim aAzi     As Single
   Dim aShDist  As Single

   For zzz = -23.4 to 23.4 step 0.5
   '==============================================
       '  zzz  is the declination here
       ' lat in this program is col-late se we must do 90-lat to get back to lat
       nAlt      = DEG(ASN(Sin(RAD(zzz))*Sin(RAD(90-lat))+Cos(RAD(zzz))*Cos(RAD(90-lat))))
       nShX      = glh/Tan(rad(nAlt))
       aAlt      = DEG(Atn(glh/(nShX  + glh)))
       aAlha     = DEG(ACS((Sin(RAD(aAlt))-Sin(RAD(zzz))*Sin(RAD(90-lat)))/( Cos(RAD(zzz))*Cos(RAD(90-lat)) ) ))
       aAhr      = aAlha/15             ' do this even though undone in aAzi to match .xls
       aAzi      = DEG(Atn(Sin(RAD(15*aAhr))/(Sin(RAD(90-lat))*Cos(RAD(15*aAhr))-Tan(RAD(zzz))*Cos(RAD(90-lat)))))
       '                 aAzi is the radial from the base of the gnomon (dial center + glh/tan(rad(lat))
       '                 aAzi is one ordinate for the asr1 point on the dial plate (i.e. direction
       aShDist   = glh + nShX              ' not used but here for completeness
       '                                  ' its glh+nShX was used above though for aAlt

       ' ABOVE calculations are for horizontal dials, but it gets us to AZI and ALT for asr times

       'aAzi & asr1ShadowDist are coordinated from the base of the nodus to the azr1 point for this decl & latitude
       '                 base of the nodus = glh / tan(radans(lat))

       ' BELOW calculations use the asr AZI and ALT for horizontal dials, but
           ' then converts to a vertical dial XXX and YYY value

       xxx = -glh * Tan(rad(aAzi))
       yyy = -xxx  * Tan(rad(aAlt)) / Sin(rad(aAzi))

       If Abs(xxx) < 1 And Abs(yyy) < 1 Then
          dcCreateCircle  0+xxx , yyy+nodusy,  0.005
       End If
   '==============================================
   Next zzz
   dcSetTextParms dcBLACK,"Ariel","Bold",0,4, 21,0,0
   dcSetLineParms dcRED,dcSOLID,dcTHIN

' end code for Islamic asr1 Jan 24 2013
End If
```

Logic in the Vertical Decliner Dial DeltaCAD macro:- BAS vDecFurniture [vDial choice 9].bas

The logic simply uses the derived azimuth and altitude (aAzi, aAlt in the previous code), and using that, draws line segments or little circles as the declination moves from -23.44 to +23.44, but only for dials facing southwest since ASR is an afternoon depiction. The base point for azimuth and altitude depiction is the base of the nodus on the sub-style line, “tx1” and “ty1”, where “ssl” is the sub style linear length:-

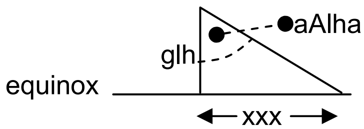
```
ssl = glh / Tan(rad(sh))
tx1 = ssl * Sin(rad(sd))
ty1 = ssl * Cos(rad(sd))
```

See also the section “AZIMUTH AND ALTITUDE ON A VERTICAL DECLINER” earlier, and See also Programming Shadows for some additional notes and logic.

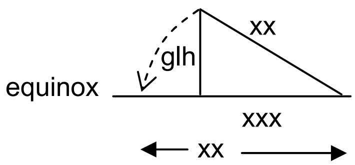
Formulae and code in the Polar Dial DeltaCAD pDial macro:-

BAS pDialFurniture [pDial choice 2].bas

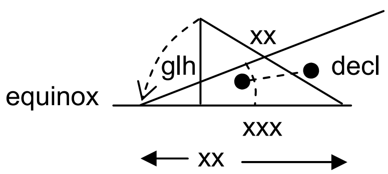
The logic is the same as the horizontal dial and vertical dial in that the ASR altitude and azimuth are derived. However, whereas the horizontal dial used a vector to go to a point, and the vertical dial used a more involved method to convert the altitude and azimuth to a point on a vertical surface, the polar dial uses the ASR local hour angle and the declination directly.



tan(aAlha) = xxx / glh
thus xxx = glh * tan(aAlha)
where xxx is the X component of the ASR point



the hypotenuse xx of the right triangle
having glh and xxx
is rotated down to the equinox line



then from that point,
a line is drawn at the declination
and where it meets the xxx position or meridian
is the local hour angle for ASR for this declination
and latitude

```
' DRAW ISLAMIC ASR1 CURVE
   Dim nAlt     As Single
   Dim nShX     As Single
   Dim aAlt     As Single
   Dim aAlha    As Single
   Dim aAhr     As Single
   Dim aAzi     As Single
   Dim aShDist  As Single
   Dim xxx      As Single
   Dim yyy      As Single
   Dim zzz      As Single

   For zzz = -23.44 to 23.44 step 0.5                    ‘ zzz  is the declination here

       ' *** General code for the ASR azimuth and altitude for this declination
       nAlt      = DEG(ASN(Sin(RAD(zzz))*Sin(RAD(lat))+Cos(RAD(zzz))*Cos(RAD(lat))))
       nShX      = glh/Tan(rad(nAlt))
       aAlt      = DEG(Atn(glh/(nShX  + glh)))
                   ' but use this for "asr2"
                   '          aAlt      = DEG(Atn(glh/(nShX + nShX + glh)))

       aAlha     = DEG(ACS(  (Sin(RAD(aAlt))-Sin(RAD(zzz))*Sin(RAD(lat)))  /  ( Cos(RAD(zzz))*Cos(RAD(lat))  )  ))
       aAhr      = aAlha/15                ' do this even though undone in aAzi to match .xls

       ' whereas the hDial logic uses the azi and alt directly
           ' and whereas the vDial converts the azi and alt to an xxx and yyy value
           ' the pDial uses the lha and decl to identify the point

       ' aAlha    is    the local hour angle in degrees
       ' zzz      is    the declination for this hour angle

       xxx = glh * Tan(rad(aAlha))     ' should be good for the XXX value
       If typ = "W" Then
          xxx = -1 * glh * Tan(rad(90-aAlha))     ' should be good for the XXX value
       End If

       ' [no sqrt function] yyy = tan(rad(zzz)) * sqrt( glh*glh +  xxx*xxx   )
           yyy = -Tan(rad(zzz)) * xxx /Sin(rad(aAlha))
       If typ = "W" Then
             yyy = Tan(rad(zzz)) * xxx /Sin(rad(90-aAlha))
       End If

       dcCreateCircle  xxx , yyy,  0.01

   Next zzz

   dcSetTextParms  dcBLACK,"Ariel","Bold",0,4, 21,0,0
   dcCreateText    0+xxx , yyy+nodusy-0.02,  0,  "ASR1"
   dcSetLineParms  dcRED,dcSOLID,dcTHIN
' end code for Islamic asr1 Jan 21 2013
```

SCRATCH (or mass) DIALS

Mass or scratch dials are part of the lore and history of sundials and thus a few notes and references are appropriate. This article accumulates information from other sources, and observations and research made by the author and a lifelong friend who tirelessly searched churches.

To the right is a picture of a well defined dial scratched into stone, hence their name. They are also called mass dials because they indicated the times for various church events (offices), one of which was mass. This photo was taken in January of 2013 in Winsham, Somerset.

These dials date between 1100 and 1600 approximately and are found in old English churches. Some have been moved over the centuries which explains why they are sometimes inside, sometimes upside down, or in places where the sun cannot cast a shadow, such as inside a porch.

DEFINITIONS: a gnomon is a device that has a shadow casting edge, called a style. The shadow's edge matching a line enables the time to be read. A nodus is a point whose shadow or point of light indicates calendar information.

Not all ancient marks on an old church are scratch dials. For example, in the 1800s marks for elevation above sea level were made looking like:

And in the Queen Elizabeth I reign, government owned property was marked like:

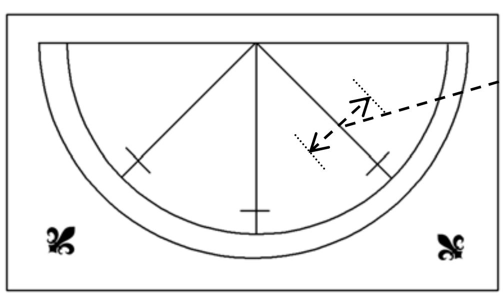
And some ancient Saxon sundials still exist which were more ornate than the scratch dial, often with ornate decorations, and within a circle or semicircle, often with two boundaries. These marked not the hours, but the center of the tides, a tide being 3 hours. The Saxon daytime tides began with Daegmael at 7:30 am to 10:30 am, the center being thus marked at 9:00 am. This was the first tide of the day. Although it is misleading to think that those times were accurate because of the nature of the dial plate, the gnomon, and wall orientation. The lines were equidistant in angle, so they were of course inaccurate unless at the equator, and no one has ever accused England of being on the equator. Secondly, it is believed the gnomon was a rod, which was thus the style or shadow casting device, and that this was horizontal, again making for inaccuracies. And even more inaccurate if the wall holding the dial was not facing true south. Other Saxon tides were Middaeg 1030 to 1:30pm, Ofanverthr from 1:30 pm to 4:30 pm and was the last tide of the day. The tides continued throughout the rest of the 24 hour period. There were thus 8 tides each of three hours. Again, those hours were not exactly compatible with ours.

Saxon dials

often two semicircles

lines marked the center
of the 3 hour tide

tides began at 7:30 am



a 3 hour tide

lines often had
a cross on them

ornate carvings
in corners or
elsewhere

The Saxon dials look ornate compared to the object of this article, namely the scratch dial. The scratch dial marked the canonical hours, and those hours marked the beginning of a church “office” which was so called because it was defined in the Official Hours. The office was in essence a church service, and originally in the 10th Century (the 900s) there were seven services

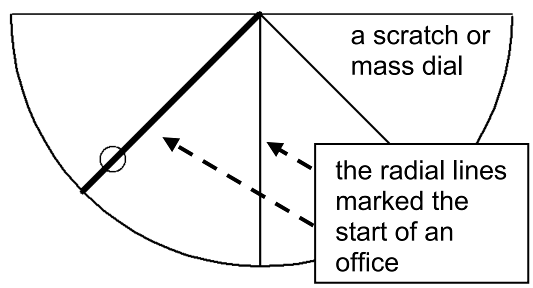
Prime	at 4am		
Matins	at 6am		
Terce	at 9am	Terce:	The third “hour”
Sext	at noon	Sext:	The sixth “hour”
Nones	at 3pm	Nones:	The ninth “hour”
Vespers	(the sunset evening service, Latin for evening)		
Nocturns	at night		

Later this range of services was restricted to those of a serious religious disposition, and the public had three services blending them into Matins, Mass, and Evensong, with mass commonly held around 9 am. Again, those hours need several grains of salt, because the nature of sundials which had equi-angular lines, horizontal styles, and walls not facing true south, were just not capable of accuracy as we understand it. But things were less hectic in those days.

The church offices and times fit well with the 24 hour day, introduced by the Romans into Britain. In fact, the octal system of the Saxons with their 3 hour tides, and the duodecimal system of the Romans coexisted in Britain. If you were of Saxon extraction you opted for tides, and those of Roman lineage used the 24 hour clock.

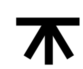
When the Normans conquered Britain, the 24 hour system became official and the tide system faded into the distant past.

How did churches mark the church offices? With rather amateur sundials is the answer, dials scratched into the walls, hence their name. And they often marked 9 am, 12 noon, and 3 pm. And those lines just happened to coincide with the lines marking the center of the Saxon tides. So a scratch dial would look very much like a Saxon dial.



However the scratch dial was much less ornate, the 9 am line for mass was often emphasized, sometimes by being heavier, sometimes with a cross on it, or a hole.

There are other markings on churches to confuse the observer. The government marks from the 1600s and the ordnance elevation marks from the 1800s could be confused with a small scratch dial, however there usually was no hole where the lines met.



Scratch dials are probably related to Saxon dials, yet inferior. It is believed their gnomons were also horizontal. The scratch dial looks similar to a Saxon dial but marked the time of a church office (from Official Hours), or service, not the middle of a tide. Coincidence makes the lines appear to be at the same angles.

I was born in Pilton Manor, Pilton, Somerset, and adjacent to the manor house is the St John The Baptist church which has two scratch, or mass, dials

Paraphrasing Dom Horne, one is:-

> cut on a buttress between the tower buttress and the south porch, some five feet three inches above the ground, with a six inch noon line. The gnomon hole is just over three inches in depth and three quarters of an inch in diameter, the dial declines S 10˚ W, the single mass line is distinct, part of the west side of the stone has been cut off, and probably moved from its original position.

And another is:-

> on the west side of the priest's door some four feet eight inches above ground, with a four inch noon line. The dial declines S 15˚ W. Above the gnomon's hole remnants of a circle are visible.

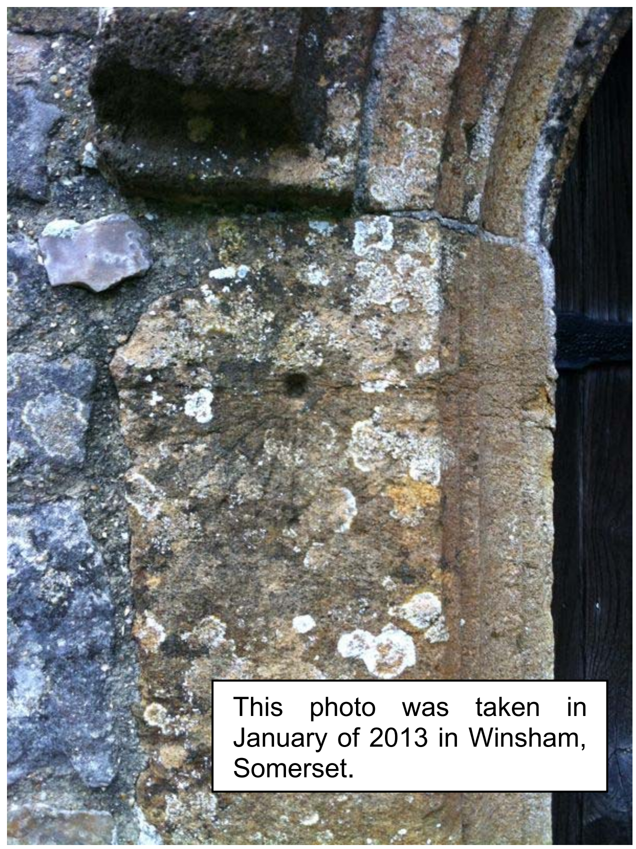
Then my parents moved to Hey Farm, Winsham, Somerset, and the Church of St Stephen has three scratch or mass dials.

Paraphrasing Dom Horne, one is:-

> on the east side of the window which is west of the priest's door and is six feet five inches above the ground. The noon line is six inches, the gnomons diameter is three quarters of an inch. The dial declines S 10˚ E, cut on very large stones that form a quoin on the west side of the priest's door

a second is:-

> on the west side of the window which is east of the priest's door, and is six feet three inches above the ground, the noon line being six and a quarter inches in length, the gnomon hole is filled in, this dial declines S 10˚ E, cut on very large stones that form a quoin on the east side of the priest's door



This photo was taken in January of 2013 in Winsham, Somerset.

and a third is:-

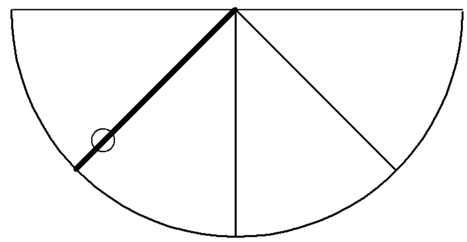
> on the west side of the priest's door and is five feet two inches above the ground, the noon line is three and a half inches, the gnomon is one half an inch in diameter, this dial declines S 10° E.

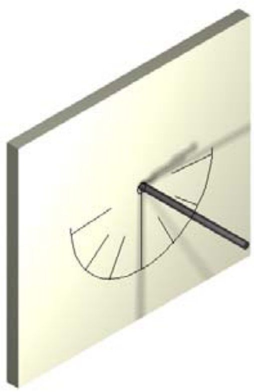
A quoin is a masonry block at the corner of a wall to strengthen it or make it appear bold and impressive.

THE BIG PICTURE OVERVIEW

Many ancient churches have scratch dials. These dials are believed to have not been scientifically designed.

Since they are vertical dials with apparently equal angles between the "hour" lines, they do not as a rule display equal hours.





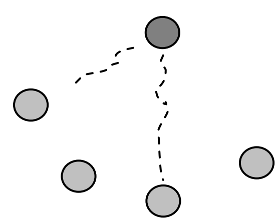
It is believed also that they do not have gnomons that were set to latitude, further reinforcing the idea of unequal hours. It is believed that the gnomons were often a straight rod of wood, stuck into a hole drilled into a rock or into the mortar holding the rocks together that form the wall.

These were empirical designs or designs the laymen unfamiliar with the art of gnomonics could employ. They were usually situated around five feet above the ground, often near an entry way to the church.

At the risk of being repetitive, the Saxon dial always had the "tides" displayed with four tides for the daylight hours, there being eight tides in a 24 hour day. The Saxon dial was ornate, and the middle of a tide was marked. The dial plate corners were often ornate.

The church offices were held at three hour intervals, so that fitted well into the Saxon tide system. The canonical hours of the church that were especially significant were around 9am and 3pm, and were often marked with a cross.

Scratch dials were more adhoc, nowhere near ornate, and some have no lines, the "hours" being only marked by holes. The 9 am line was often enhanced, others less so.



There is no direct evidence of how the scratch dial was built, by whom, how it was used, nor what the gnomons were like. Manuscripts describe old sundials, but not the scratch dial. Much of our knowledge is supposition.

WHAT WERE THE GNOMONS (STYLES, OR SHADOW CASTING DEVICES) LIKE

Gnomons were of wood, verified by the extraction of ancient wood from the style or gnomon holes.

Some researchers (Dom Ethelbert Horne) suggest that the gnomons or styles were simple horizontal rods, this presumably based on the difficulty of bending a wooden rod to latitude, and also presumably based on the need for a very simple method of making scratch dials by laymen not familiar with the concept of latitude.

Some researchers (George F. J. Rosenberg) suggest that since there was one morning service for lay people, yet several “hour” lines, that horizontal gnomons were used, but the several “hour” lines were actually for the same service, one marking the time for mass in the winter, one in the summer, and one for the spring or autumn, times close to the equinoxes.

Some researchers (Arthur Robert Green) suggest that the gnomons had to be sloped at latitude, this theory being based on the significant variations of time for a church event, service, or office, such as mass.

There is no direct evidence that gnomons were either horizontal or sloped at latitude. There is reasonable evidence that gnomons were usually wood, and there is reasonable evidence that these dials were constructed by people who were not skilled in the art of gnomonics. There is no written documentation on how these scratch dials were made, by whom they were made, nor how they were used. To date, conjecture is the foundation of scratch dial knowledge, pieced together with reasonable evidence, reasonable assumptions, and some conjectured theory.

WHAT DID SCRATCH DIALS LOOK LIKE AND WHERE ARE THEY FOUND

Scratch dials are often found on south or almost south facing walls, near the main door or the priests door, on quoins (corner pieces), or visible from people walking along the path approaching the church.

Typical sizes of scratch dials were found to be:-	gnomon hole	½ to 1 inch
	dial plate	1 ½ to 10 inches
	height	4 to 7 feet above ground

And the kinds of scratch dials were classified by Dom Horne in the early 1900s into twelve styles. Others have classified them into two groups. These are arbitrary classifications, those who made the scratch dials probably did not think of such things.

Almost all scratch have an emphasized noon line, typically vertically down from the gnomon or style hole. A “mass” line was also usually emphasized, being around 9:00 am, since mass was a main purpose of the dial. This would be around 45 degrees on the morning, or left, or west side of the dial. And quite a number have a similar 45 degree line on the afternoon, east, or right side of the dial also. Hence the caution not to confuse the government ownership or ordnance elevation marks.

Many dials may have been mutilated, sometimes by vandals, sometimes be people wishing to “complete” a dial, and these confuse the original purpose. Original mass and noon lines were often emphasized, subsequent lines were often less so, and may be incomplete, or not straight. Lines above the gnomon hole were often not original, however their use and manner of making is a matter of conjecture. Sometimes a copy dial or two exist, they may have been experiments, often close to the final dial. Some believe different priests, changes to service times, of growing yew tree shadows were the cause of copy dials, or at least the cause of a church having multiple scratch dials.

WHAT DID SCRATCH DIALS PRESUMABLY INDICATE

It is assumed that scratch dials do not show equal hours because horizontal style in a vertical dial at England's latitude would not have equi-angular lines. Also, walls were sometimes not facing true south. Any one of those problems would make for unequal hours. It is asserted that their sole purpose was to mark times of church services. Such services being mass, vespers, and the like. Some common offices or events were marked by a line, sometimes by a hole, and may have signaled:-

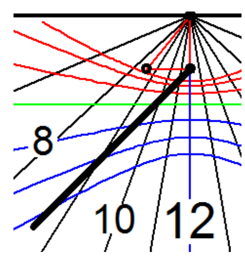
	summer		winter
	|		|
"Terce"	0800	0900	1000
"Sext"		1200	
"None"	1600	1500	1400

but these varied with the season as the gnomon was believed to have been horizontal and not set at latitude

In the 10th Century (the 900s) there were seven services,

Prime	4am	Matins	6am	Terce	9am	Sext	at noon
Nones	3pm	Vespers		(sunset evening service, Latin for evening)			
Nocturns		at night					

Later this range of services was restricted to those of a serious religious disposition, and the public had three services blending them into Matins, Mass, and Evensong. At the time many scratch dials were marked England was still Roman Catholic, the Church of England did not exist until King Henry VIII appeared on the scene.



The horizontal style meant for example that the 45˚ line for the nominal 9am and 3pm offices was closer to noon in winter than the summer, allowing morning and afternoon farming tasks to be completed. A gradual daylight saving time system as opposed to the one hour jump of modern times! To the right, the "9am" 45˚ line is around 11am in winter, 10am on the equinox, and just before 9am in the summer. The "3pm" office was similarly affected. Very appropriate for an 8 then 12 then 16 hour daylight seasonal range!

HOURS AND TIDES

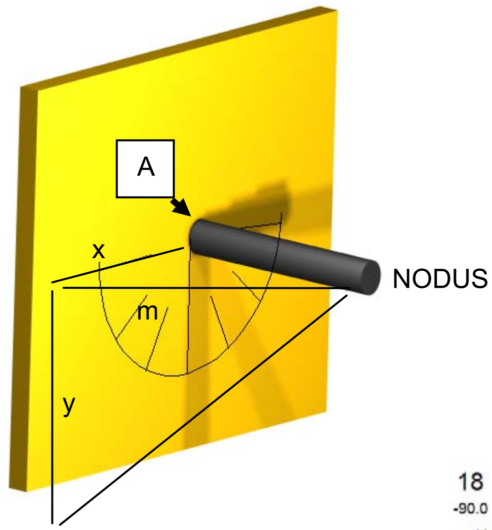
The Saxons divided the solar day into 8 tides each of 3 hours, a line with a cross on it often marked the middle of a tide and this is also seen on many scratch dials. However, the canonical hours which were in the middle of a tide, also often had crosses on them. The Saxons built ornate sundials, whereas scratch dials were seldom if ever ornate. Caution should be made when comparing a Saxon dial with a scratch dial, the presence of a cross emphasizing a line may not differentiate between a Saxon and a scratch, its location or ornate nature may be a better indication.

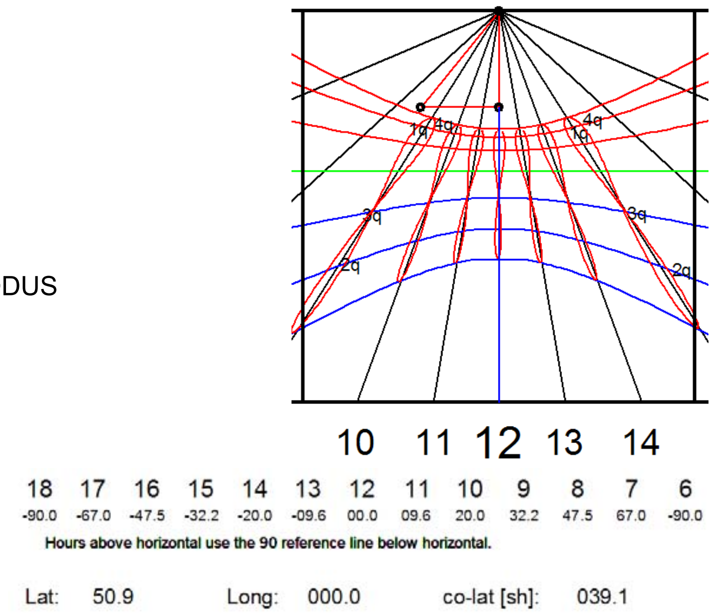
Later, the Normans brought the daytime length of 12 hours, however the Romans also introduced the 12 hours of daytime to England. With the Normans, however, the Saxon tides were obsoleted.

The Roman and Norman 12 hour daytime and the Saxon 8 tides in the solar day co-existed, those of a Saxon heritage opting for their tides, while those descended from the Romans continued with their 12 hour daytime, and 12 at night. Many dials of those times appear to have both indications of the tides, as well as the church or canonical hours.

Some interesting trivia: clocks started to be used in the 1200s, Westminster acquiring one in 1288, often the clock face was inside the building, it was not until the 1300s that some clock faces were outside the building.

SCRATCH OR MASS DIALS CONSIDERING ANGLES FROM A HORIZONTAL STYLE

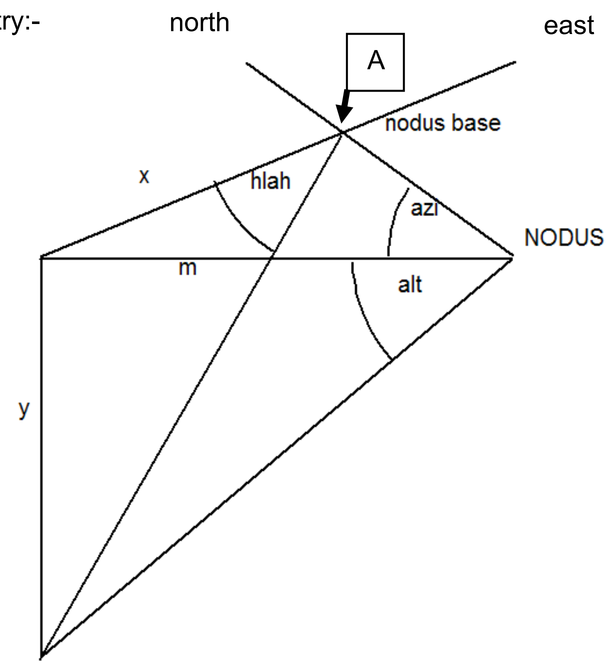




Consider a dial plate for latitude 50.9 as in Somerset, England and for local apparent time (no longitude consideration). The top left figure represents a horizontal gnomon, the top right figure represents a true south vertical dial with a style at latitude, aligned with the meridian.

ISSUE 1: the gnomon's shadow angle from vertical for 9 am

The above can be derived with trigonometry:-



sin(azi) = x / m
so... x = m * sin(azi)

tan(alt) = y / m
so... y = m * tan(alt)

and since

hlah = hour line angle:horizontal

tan(hlah) = y / x
= m * tan(alt) / m * sin(azi)
= tan(alt) / sin(azi)
so
hlah = atan(tan(alt) / sin(azi))

so hlav = hour line angle:vertical
= 90 – atan(tan(alt)/ sin(azi))

and azimuth and altitude are derived from the formulae in appendix 8.

When the formula: hlav.scratch = 90 – atan(tan(alt) / sin(azi))

is entered into a spreadsheet such as: XLS bk3 sup scratch dial.xls

and latitude set to 50.9 and declinations on +23.44 and 0 and –23.44 are used for the 0900 hour the results are:-

RESULTS		summer	solstice	
LATITUDE	DECL	HOUR	HLAV	HLAH
50.90	23.440	9.00	42.11	47.89

RESULTS		equinox		
LATITUDE	DECL	HOUR	HLAV	HLAH
50.90	0.000	9.00	57.76	32.24

RESULTS				
LATITUDE	DECL	HOUR	HLAV	HLAH
50.90	-23.440	9.00	81.20	8.80

The summer hour line for 0900 would be 42.11º

The equinox hour line for 0900 would be 57.76º

The winter hour line for 0900 would be 81.20º

a net variation of 81.20 – 42.11 or 39.09º

Green uses angles from horizontal “HLAH”

And these calculated figures agree fairly well with Green’s book page 85. Some researchers suggest this variation is excessive, and thus proposed that gnomons must have been bent to latitude and aligned with the north south meridian in order to display more consistent times for mass.

ISSUE 2: the gnomon’s horizontal shadow hits what hour line during the year

And let us consider on what hour a shadow will fall when the shadow from point “A” is horizontal, this is sunrise or sunset, so the altitude of the sun is 0º

			Local Apparent Time rise h.mm	 set h.mm
For latitude 50.9	winter solstice	declination = - 23.44	8.08	15.51
	equinox	declination = 0	6.00	18.00
	summer solstice	declination = + 23.44	3.51	20.09

Since sunrise happens when the sun’s altitude is 0, then the horizontal hour line on a scratch dial with a horizontal style will be sunrise. It will appear at the equinoxes and winter, but not at the summer solstice because the azimuth places it behind the dial plate. Even so, just between the winter solstice and the equinox there is a 2 hour 8 minute variation which is significant.

CONCLUSION

These excessive variations have caused some researchers to assert that scratch dial gnomons or styles were not horizontal as others have suggested, but rather must have been bent at latitude and aligned with the true north south meridian. That would be the only way to avoid significant variations of the time for mass as the seasons vary. It should be observed that there is no direct evidence of styles being set at latitude, and equally, no direct evidence of them being horizontal. Further, even a gnomon at a latitude of say 50.9º would not generate the hour line of 45º at 9 am, it would be at 32.2 degrees. The best solution is to enjoy these dials with their charm of the ages.

RESOURCES:

Primitive Sundials or Scratch Dials Dom Ethelbert Horne, 1917	Somerset
Classification of church scratch-dials (8 pages) T. W Cole (1935)	
Scratch-Dials and medieval church sundials (16 pages) T. W Cole (1938)	all counties
Sundials, Incised Dials or Mass-Clocks Arthur Robert Green, 1978	Hampshire
Scratch Dials, Sundials and Unusual Marks on Herefordshire Churches Richard and Catherine Botzum, 1988	Hereford
Time in Rutland Robert Ovens & Sheila Sleath	Rutland
The British Sundial Society BULLETIN, eg volume 23, iii etc articles by Chris H. K. Williams	
British Sundial Society http://www.sundialsoc.org.uk/massdials.php	

KEY POINTS TO REMEMBER:

Saxon Dial	Ornate dial, did not necessarily relate to a church service or office
Scratch Dial	Not ornate, scratched onto a stone or rock in a wall
Mass Dial	A scratch dial that was obviously used for church services or offices
Scientific dial	A dial with equal hours, such as in modern use
Transitional dial	A scratch or mass dial but having some additional markings such as Roman Numerals
Times of Worship	While mass dials were part of the Roman Catholic way of life, other religions had similar uses for sundials, for example Islam and the times for prayer.
Geographical area	While mass dials were common on old English churches, they have been found on the continent also. And of course the Islamic dials for the calls to service were found in Muslim countries.

The British Sundial Society offers a form to document these dials and a set of explanations, the form is reproduced here:- http://www.sundialsoc.org.uk/massdials.php

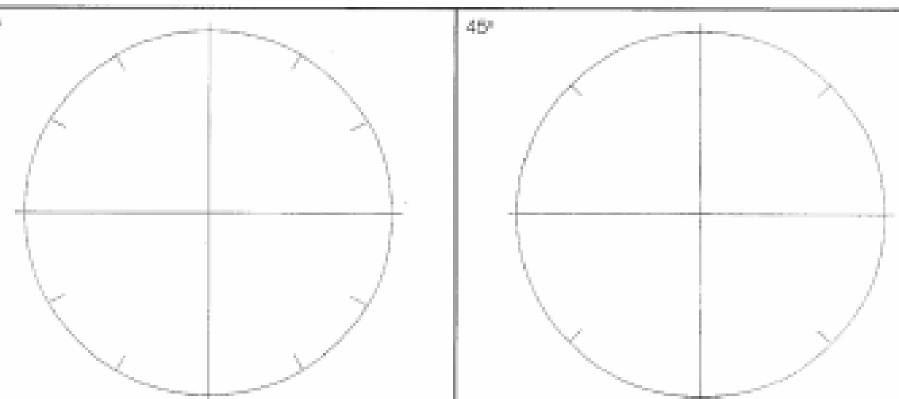
BRITISH SUNDIAL SOCIETY **Mass Dial Record**

MDR No.

PLACE			COUNTY			
CHURCH / OWNER	DEDICATION					
(AGE OF CHURCH)	LANDRANGER		O.S. REFERENCE			
ON THIS BUILDING (of)	HEIGHT ABOVE GROUND		ASPECT			
POSITION (No.)			CONDITION			

DESCRIPTION (ANGLES OF RADII)		NUMBERS R/A	CIRCLES	RADII	POCKS	
		OTHER FEATURES				
DIAL RADIUS	GNOMON	GNOMON HOLE :	MORTAR LINE	DIA.	DEPTH	
FURTHER NOTES (LAT. LONG.)			ALSO :	SUN DIAL(S)	SAXON DIAL	

30° 45°



REFERENCES			
RECORDER	BSS	DATE SEEN	PHOTO

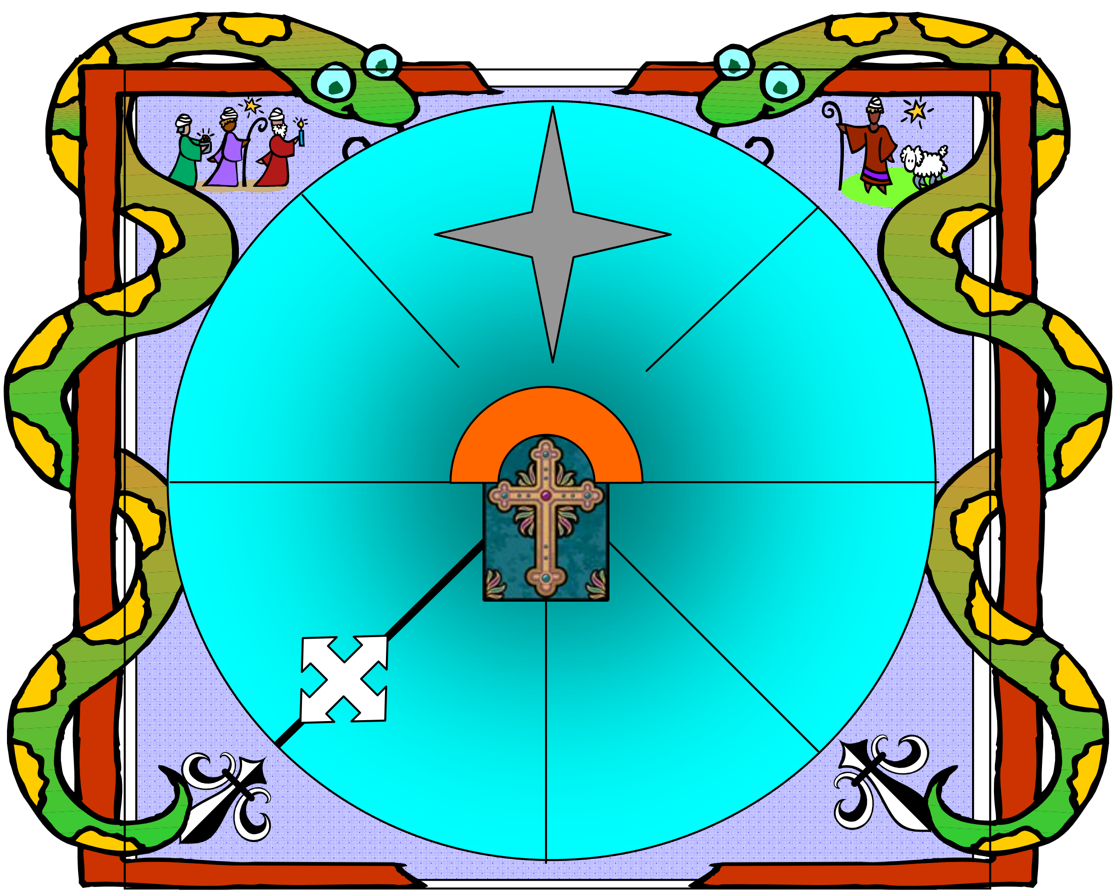
FUN WITH SCRATCH DIALS

Scratch dials have a charm of their own. Partly for me because the church where I was christened, then the church we went to as I was growing up and where my brother was buried, have them. Partly because they are aged, old, and covered with the aged lichen of time, and like me come from a time past. Partly because they are part of the lore of sundials in general. Being an introvert, I share a quiet life with them. Who knows.

The general theory behind their construction, their use, and whether they had a horizontal gnomon with a few "hour" lines, a horizontal gnomon with separate "hour" lines for the seasons, several dials copied for the seasons, a sloping gnomon, they are all conjecture. No documents seem to have been uncovered that address who made them, when, how, and their method of use. So it seems reasonable that if conjecture is the mainstay of their use, and construction, why not expand conjecture about how they actually looked in daily life.

Chris Williams in the BSS (British Sundial Society) Journal did just this. And since we accept conjecture for the cold hard facts of design and usage, why not do the same and see what life was like in the churches when these dials were in use.

And when these dials were in use it was a Roman Catholic Church, and those churches were not the austere protestant buildings of puritanical simplicity, places where art and color were removed and abhorred. No, the churches were awash in art and color. It seems logical that if anything that could be painted, was, then why not the same for a scratch dial? Of course, there would be no traces of such decoration after the centuries have passed them by, so what is unreasonable about assuming artwork on the reasonable sized scratch dials would match the concepts and colors of the decorative religious work inside, art work by paint rather than sculpture? The only argument against it would be that these were small outside items and exposed to the elements.



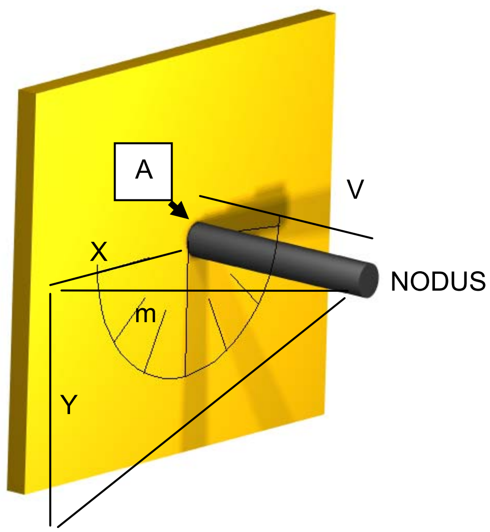
So, advancing to the next item of interest, consider the nodus of a horizontal gnomon's style. The formula for azimuth and altitude of the sun is:-

ALTITUDE: The sun's altitude is its angle when looked at face on

alt = ASIN(SIN(dec) * SIN(lat) +
COS(dec) * COS(lat) * COS(lha))

AZIMUTH: azi = ATAN(SIN(lha) / ((SIN(lat) * COS(lha))
– (COS(lat) * TAN(dec)))

where the sun's declination is "dec", and the latitude is "lat", and the sun's local hour angle is "lha". And given the altitude and azimuth then the "X" and "Y" coordinates of the shadow tip are:-



tan(azimuth) = X / V
so
X = V * tan (azimuth)
next
set V = glh [gnomon linear height/length] = 1
then
X = tan(azimuth)

now, derive "m" for the next phase
sin(azimuth) = X / m
so
m = X / sin(azimuth)

Y = tan(altitude) * m
so

$$Y = \frac{\text{tan(altitude)} * X}{\text{sin(azimuth)}}$$

If the above is correct then we should be able to derive a vertical dial (true south) hour line angle by:

tan(hour line angle) = change in X / change in Y

where the change is for two different solar declinations

This is demonstrated in the spreadsheet: XLS bk3 sup scratch dial.xls
worksheet: X and Y

the raw X and Y values are shown on the next page for a latitude of 50.9 N (a good latitude for Somerset, England)

AM	decl =	for lat	50.90	raw X and Y values	
hour	-23.44	Azimuth	Altitude	x	y
12.0		0.00	15.7	0.00000	#DIV/0!
11		14.20	14.5	0.25299	0.2665744
10		27.87	11.1	0.52881	0.2217968
9		40.70	5.8	0.86003	0.1331647
8		52.63	-1.1	1.30926	-0.0319455
7		63.85	-9.1	2.03655	-0.3652523
6.0		74.71	-18.0	3.65710	-1.2304997

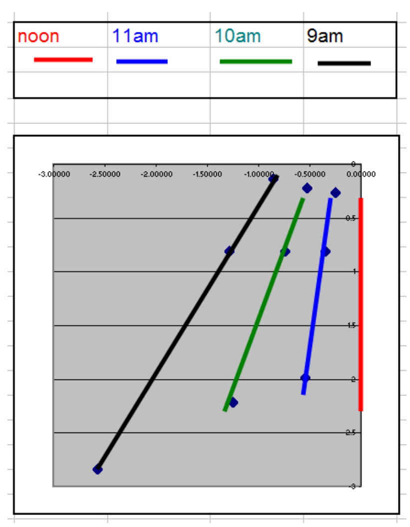
	decl =				
	0	Azimuth	Altitude	x	y
12.0		0.00	39.1	0.00000	#DIV/0!
11		19.05	37.5	0.34527	0.812678
10		36.65	33.1	0.74396	0.812678
9		52.19	26.5	1.28858	0.812678
8		65.87	18.4	2.23189	0.812678
7		78.25	9.4	4.80906	0.812678
6.0		90.00	0.0	#########	0.6306758

	decl =				
	23.44	Azimuth	Altitude	x	y
12.0		0.00	62.5	0.00000	#DIV/0!
11		28.53	60.2	0.54355	1.9859914
10		51.44	54.1	1.25428	2.2141797
9		68.73	45.9	2.56843	2.8420051
8		82.46	36.7	7.55809	5.688535
7		-85.70	27.3	-13.30749	6.8843271
6.0		-74.71	18.0	-3.65710	1.2304997

Then excluding noon (division by 0), and excluding 8am, 7am, and 6am as the sun is below the horizon in the winter, then that leaves 11am, 10am, and 9am which is more than enough to calculate the X difference and the Y differences. And next calculate the angle between X,Y for a given time for one declination, and again for a second declination, and from that derive the vertical dial's hour line angle. Below left are the X and Y pairs for the winter solstice first, the equinoxes second, the summer solstice third.

These values can be plotted using X,Y scatter in Excel.

x	y
-0.25299	-0.266574
-0.52881	-0.221797
-0.86003	-0.133165
-0.34527	-0.81268
-0.74396	-0.81268
-1.28858	-0.81268
-0.54355	-1.98599
-1.25428	-2.21418
-2.56843	-2.84201



Of course, it must be remembered that the charts Excel produces do not retain their aspect ratio. So, the hour lines angles are now calculated below from the X,Y differences between the two solstices:-

hla from alt/azi differences

hla=atan((y2-y1)/(x2-x1))	
12.0	#DIV/0!
11.0	9.591741019
10.0	20.00761966
9.0	32.23863845

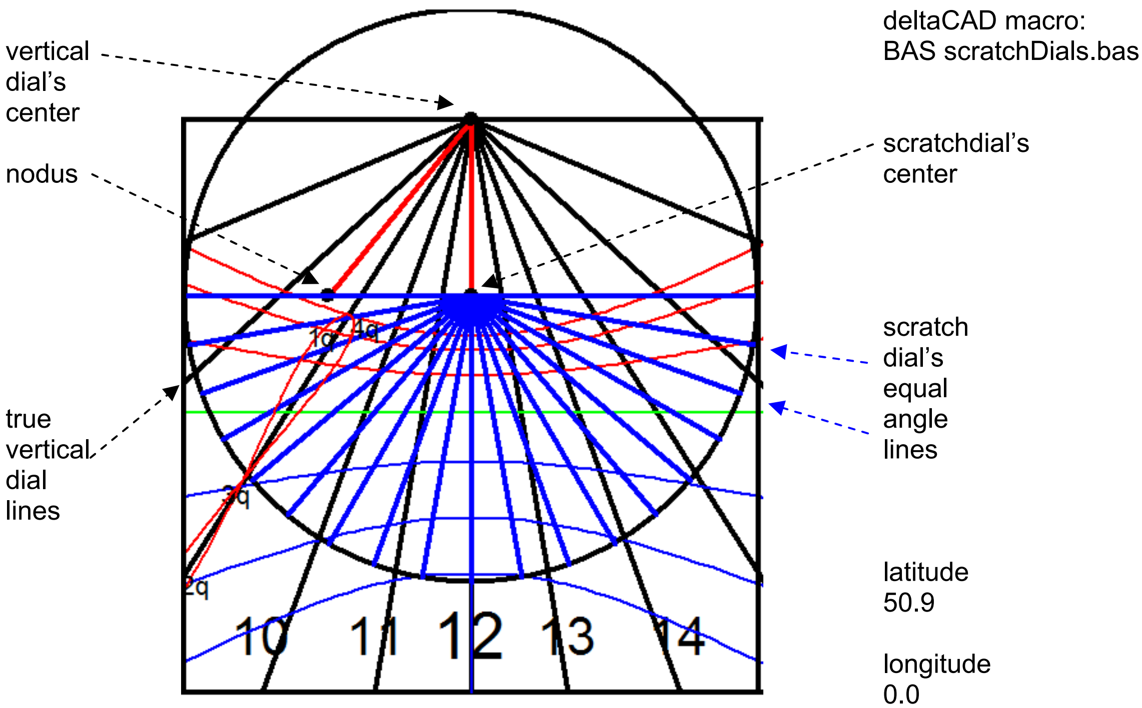
Standard v-dial formula

hla=atan(sin(90-lat)*tan(ha))	
12.0	0
11.0	9.591741
10.0	20.00762
9.0	32.23864

As they say, the swiftness of the eye defeats the mind. They match!

What these last three pages demonstrate is nothing more than the obvious, namely that the tip of the gnomon is a nodus, and that for the same hour at different times of the year, ignoring the EOT (equation of time), a straight line will be drawn by that nodus, and that line will have an hour line angle derives from the altitude and azimuth, that matches the hour line angle derived from the sun's hour angle.

Superimposing a true vertical dial with a dial center, and the scratch dial whose lines radiate from the base of the nodus, the hole holding the gnomon, in other words, one then sees the following:-



The whole point of the above picture, apart from demonstrating that some people have too much time on their hands, is to show the obvious, namely that had the concept of equal hours been of significance, as well as a method of determining the time, while eliminating the equation of time as a factor, then a scratch dial could obviously have been an accurate time keeper. And had there been an accurate time piece, and set to noon each day using the noon line, then the equation of time would in essence have been removed from the equation.

The only point of this section has been purely to tie things together, and have some fun.

DECLINATION CONFUSION

The author of Illustrating Time's Shadow, namely me, has received countless emails from pilots, navigators, and the like, demanding that the term magnetic declination not be used, it just confuses the issue. And they have my undying support. However the fact is that dialists use the term declination in three ways, and nothing I say or do will ever change that. So what does declination mean? There are three declinations in common usage, and they have little if anything in common.

1. Solar declination
2. Magnetic declination
3. A wall or vertical surface declination

THE DECLINATION OF THE SUN, solar declination, is the angle the sun makes as it orbits the Earth's equator. The sun is – 23.44 degrees at the December solstice, + 23.44 degrees at the June solstice, and 0 degrees at the spring and autumn equinoxes. The formula for solar declination is simple:-

```
DEGREES(0.006918 - 0.399912*COS(((2*3.1416*(jd-1)) / 365)) + 0.070257*SIN(((2*3.1416*
        (jd-1)) / 365))  - 0.006758*COS(2*((2*3.1416*(jd-1)) / 365)) + 0.000907*SIN(2*((2*
        3.1416*(jd-1)) / 365)) - 0.002697*COS(3*((2*3.1416*(jd-1)) / 365)) + 0.00148*SIN(3*
        ((2*3.1416*(jd-1)) / 365)))
```

```
DEGREES = (23.45*sin(radians(0.9678(jd-80))))
```
alternative formula agrees within half a degree

The above two alternative formulae are as they might be seen in a spreadsheet.

To the right is a simple summary table of the solar declination.

You may notice that the above formulae use "jd" which means Julian Day. In the context above, that means the day of the year starting at 1 ending at 365 or 366. A simple summary table of Julian days appears below.

	DECLINATION		
	5th	15th	25th
JAN	-22.7	-21.3	-19.2
FEB	-16.2	-13.0	-9.4
MAR	-6.3	-2.4	1.5
APR	5.8	9.5	12.9
MAY	16.0	18.7	20.8
JUN	22.3	23.3	23.4
JLY	22.9	21.7	19.8
AUG	17.2	14.3	11.0
SEP	7.1	3.3	-0.5
OCT	-4.4	-8.2	-11.8
NOV	-15.5	-18.3	-20.6
DEC	-22.3	-23.2	-23.4

	JULIAN DAYS		
	5th	15th	25th
JAN	5	15	25
FEB	36	46	56
MAR	64	74	84
APR	95	105	115
MAY	125	135	145
JUN	156	166	176
JLY	186	196	206
AUG	217	227	237
SEP	248	258	268
OCT	278	288	298
NOV	309	319	329
DEC	339	349	359

The sun's declination is used for declination curves or calendar curves, and for length-of-day curves. Because there are two dates for each solar declination, except at the solstices, below are some good dates near the 21st of the month to use with symmetrical declinations.

23.44˚ Jun, Dec	20˚ good or 19˚, or 18˚ Jan, May Jly, Nov	10˚ good or 11˚ or 12˚ Feb, Apr Aug, Oct	0˚ Mar, Sep

MAGNETIC DECLINATION is what is in essence called compass variation. Magnetic compasses point to the magnetic pole which moves and is not at the north pole. The compass variation is called magnetic declination. In essence they are one and the same. Two mnemonics help determine what a compass should read, or what a true heading or bearing would be.

TEMPE	true equals magnetic PLUS an easterly variation/declination
MITME	magnetic is true minus an easterly variation/declination

The western part of the USA has an easterly variation, the east part has a westerly deviation. The variation can vary by about one degree in 10 years, for example, three decades of Phoenix Arizona magnetic variation/declination indicate:-

Sept 25, 2012	**10.85** East
Sept 25, 2002	11.8 East
Sept 25, 1992	12.46 East

Web sites offer the current magnetic declination or compass variation, same thing, and charts are published of the isogonic lines, lines sharing the same magnetic declination or compass variation.

There is also a term "magnetic deviation" which refers to local minerals or metals that can cause the compass needle to deviate from the magnetic north. This is corrected in ship and aircraft compasses by introducing an equal and opposite deviation, the picture to the right shows two spheres that can be adjusted. The adjustment process is called swinging the compass.

However, that is not feasible when using a compass in the wild. Compasses can be deviated by the obvious, namely garage doors, automobiles, and rebar in sundial columns.

Not so obvious is that the column itself even with no rebar can cause a deviation. For example, kiln fired bricks when cooling can establish a magnetic alignment that can cause a magnetic deviation. The moral of this is that considerable care must be used when using a magnetic compass to ascertain true north.

THE DECLINATION OF A VERTICAL SURFACE, SUCH AS A WALL, is the angle by which it is displaced from true north or true south. Thus a declination of S 45 W would be a wall that was displaced from the south towards the west by 45 degrees.

While Illustrating Time's Shadow covers many ways of determining this, a summary s in order along with cautionary notes.

COMPASS ~ The obvious method is to use a magnetic compass. However several considerations must be considered. First, the magnetic declination or variation must be known. And second local magnetic disturbances must be considered. Once magnetic north is found, true north is deduced, and a protractor then used to determine the vertical surface declination.

ASTRO COMPASS ~ An astro compass is often accurate to within a degree or two, but requires an accurate clock, an accurate longitude, and latitude, and solar declination, along with an accurate equation of time. Once true north is found, the astro compass scale will determine the vertical surface declination.

EQUAL ALTITUDE ~ Here the altitudes of the sun are recorded on a surface that has lines or circles to locate equal sun altitudes. A line from the base of a vertical shadow casting device to the mid point of equal altitudes is true north. Once true north is deduced a protractor is then used to determine the vertical surface declination.

SOLAR NOON ~ Armed with longitude, the astronomically accurate equation of time, and an accurate clock, a noon shadow from a vertical shadow casting device will provide true north. Once true north is deduced a protractor is then used to determine the vertical surface declination.

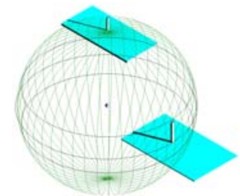
POLARIS ~ should you venture out at night, a line from you the observer to the north star will provide true north within a degree or so. Once true north is deduced a protractor is then used to determine the vertical surface declination. This does not work when Polaris is not visible, and thus useless in the southern hemisphere.

AZIMUTH MEASUREMENT ~ Since the sun's azimuth is known on any day, armed with longitude, the astronomically accurate equation of time, and an accurate clock, a shadow on a vertical board can be tracked, the sun's azimuth determined by those shadows and compared to the calculated azimuth for each shadow. The difference provides the vertical surface declination. Several readings are made, and the Illustrating Time's Shadow main spreadsheet allows averaging to provide a good declination.

Chapter 6 of Illustrating Time's Shadow covers these methods in great detail.

The three declinations have nothing in common, except that they indicate a difference in degrees of something.

THE OLD HOUSEWIFE'S TRICK



How do you adjust a sundial for the longitude difference, how do you adjust it for the equation of time, EOT.

A number of books talk of the old housewife's trick of rotating the sundial until it shows the correct time.

First, the correct way to do things. Atkinson's Theorem says that you may rotate an hour angle sundial around its style to effect a shift in time. This works.

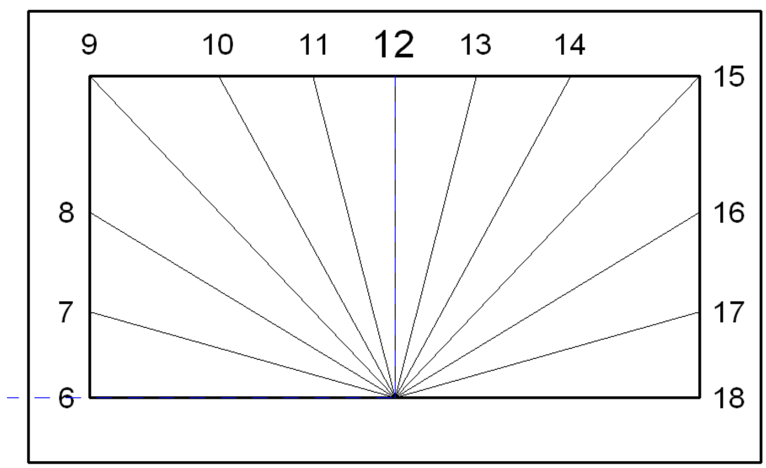
However, this should not be confused with what to do when a sundial is designed for a different latitude, when it is tilted to the north or south until the style now is set at latitude for the current location.

But what about rotating the plate.

For an equatorial dial (one that is a vertical non decliner on the equator), or an armillary dial, the hour line angles are 15 degrees apart and thus, being linear, may be rotated to effect a linear change of time. This may correct for longitude differences from the legal time meridian, and if done regularly, such as every 10 days, can also adjust for the equation of time. The rotation is done around the style.

The old housewife's, or is it wives, trick does not do that, it rotates the dial as it rests horizontally.

Below is a horizontal dial for the north pole, latitude 90 degrees.



Hour and hour line angle H-DIAL

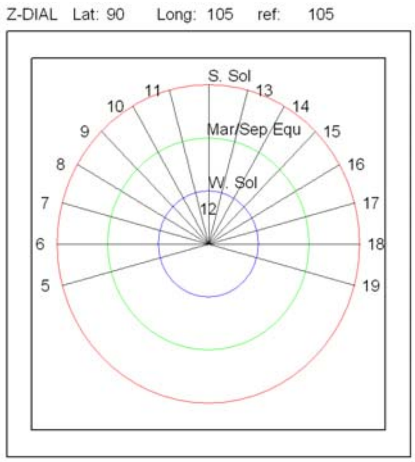
6	7	8	9	10	11	12	13	14	15	16	17	18
-90.0	-75.0	-60.0	-45.0	-30.0	-15.0	00.0	15.0	30.0	45.0	60.0	75.0	-90.0

Hours below horizontal use the 90 reference line below horizontal.

Lat: 90.0 Long: 105.0

Here, a horizontal dial on the north or south pole has hour line angles of 15 degrees. Think about it, a plate on the pole is an azimuth dial

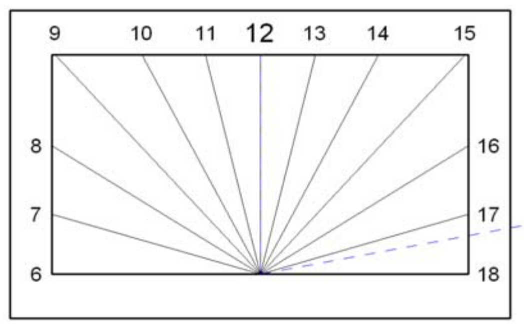
Here is an azimuth dial for the pole, it is the same as a horizontal dial. The learning from this is that an azimuth dial uses one component of an hour angle dial. Similarly with an altitude dial. Or, another learning is that an hour angle is made up of the altitude and azimuth components.



Assumes spline=azi/decl as approx

So, at the north pole, rotating a horizontal dial can be used to correct for longitude differences from the legal time meridian, or, if done frequently, such as every 10 days, can also be used to correct for the sun's variation, called the equation of time, or EOT.

At latitude 80 this works well.



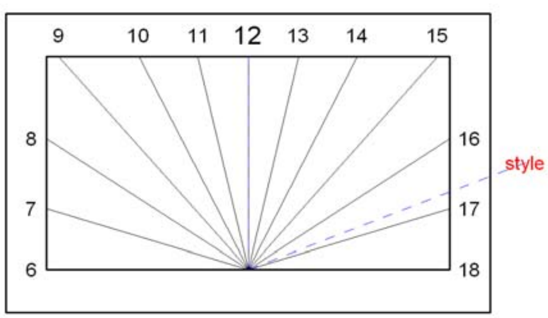
Hour and hour line angle H-DIAL

6	7	8	9	10	11	12	13	14	15	16	17	18
-90.0	-74.8	-59.6	-44.6	-29.6	-14.8	00.0	14.8	29.6	44.6	59.6	74.8	-90.0

Hours below horizontal use the 90 reference line below horizontal.

Lat: 80.0 Long: 105.0

At latitude 70 it still is usable.



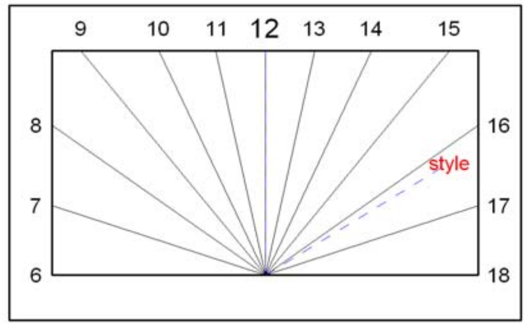
Hour and hour line angle H-DIAL

6	7	8	9	10	11	12	13	14	15	16	17	18
-90.0	-74.1	-58.4	-43.2	-28.5	-14.1	00.0	14.1	28.5	43.2	58.4	74.1	-90.0

Hours below horizontal use the 90 reference line below horizontal.

Lat: 70.0 Long: 105.0

At latitude 60, the hour lines angles and clearly not linear, but the trick is close enough.



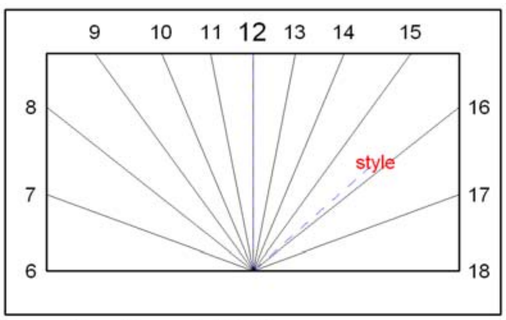
Hour and hour line angle H-DIAL

6	7	8	9	10	11	12	13	14	15	16	17	18
-90.0	-72.8	-56.3	-40.9	-26.6	-13.1	00.0	13.1	26.6	40.9	56.3	72.8	-90.0

Hours below horizontal use the 90 reference line below horizontal.

Lat: 60.0 Long: 105.0

At latitude 50 the angles become less usable.

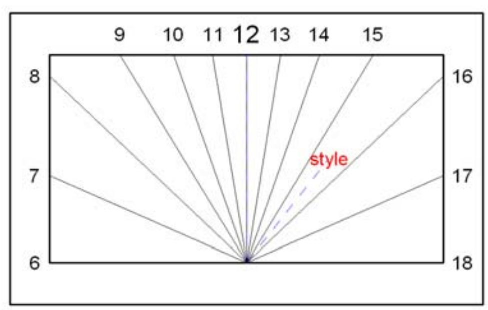


Hour and hour line angle H-DIAL

6	7	8	9	10	11	12	13	14	15	16	17	18
-90.0	-70.7	-53.0	-37.5	-23.9	-11.6	00.0	11.6	23.9	37.5	53.0	70.7	-90.0

Hours below horizontal use the 90 reference line below horizontal.

Lat: 50.0 Long: 105.0



Hour and hour line angle H-DIAL

6	7	8	9	10	11	12	13	14	15	16	17	18
-90.0	-67.4	-48.1	-32.7	-20.4	-9.8	00.0	09.8	20.4	32.7	48.1	67.4	-90.0

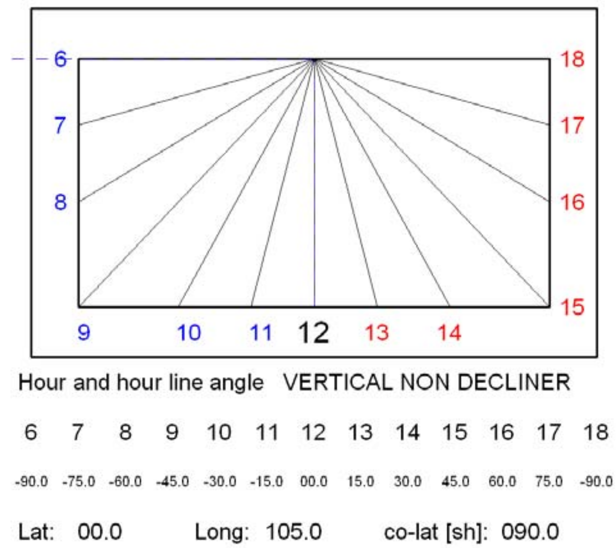
Hours below horizontal use the 90 reference line below horizontal.

Lat: 40.0 Long: 105.0

At latitude 40 the system breaks down.

And of course, at the equator, the horizontal dial is a polar dial because its dial plate parallels the polar axis.

However, a vertical dial at the equator, has 15 degree hour line angles, but that is not of interest to us here.



SUMMARY

The old housewife's trick of turning a horizontal dial horizontally works perfectly at the pole, and work fairly well down to latitudes of about 50 degrees or a bit closer to the pole. This is true for the south hemisphere also.

WHAT THE OLD HOUSEWIFE'S TRICK CAN CORRECT FOR

You can correct for longitude.

You can do regular corrections for the equation of time.

WHAT YOU CANNOT CORRECT FOR USING THIS TRICK

A dial plate designed for a different latitude can be corrected by tilting to the pole or equator, see the main book Illustrating Time’s Shadow.

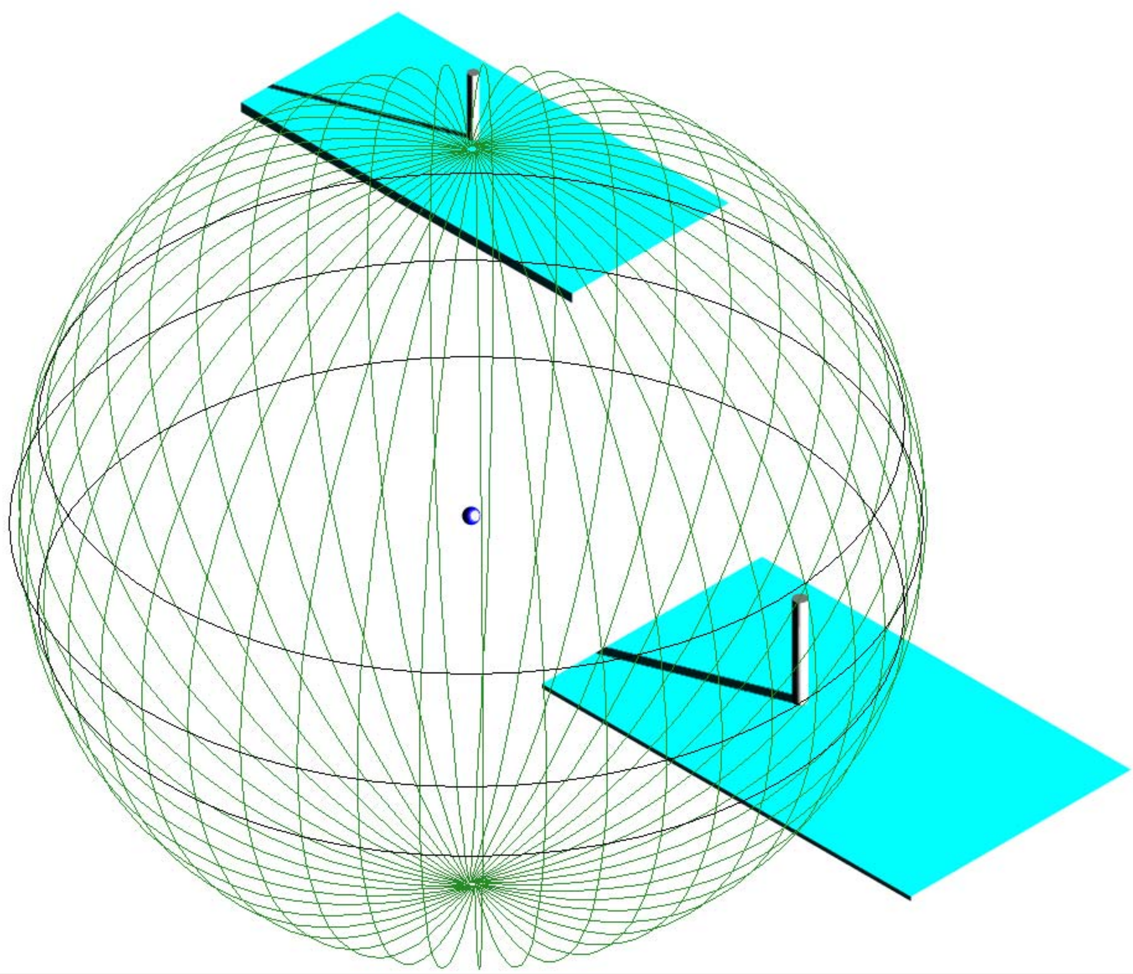
And you can use the spreadsheets also to determine the design latitude for an unknown dial:-

00aDroidUnknownDial.xls	[in the smartPhone section]
illustrating-shadows.xls	[the master spreadsheet]

PICTORIALLY

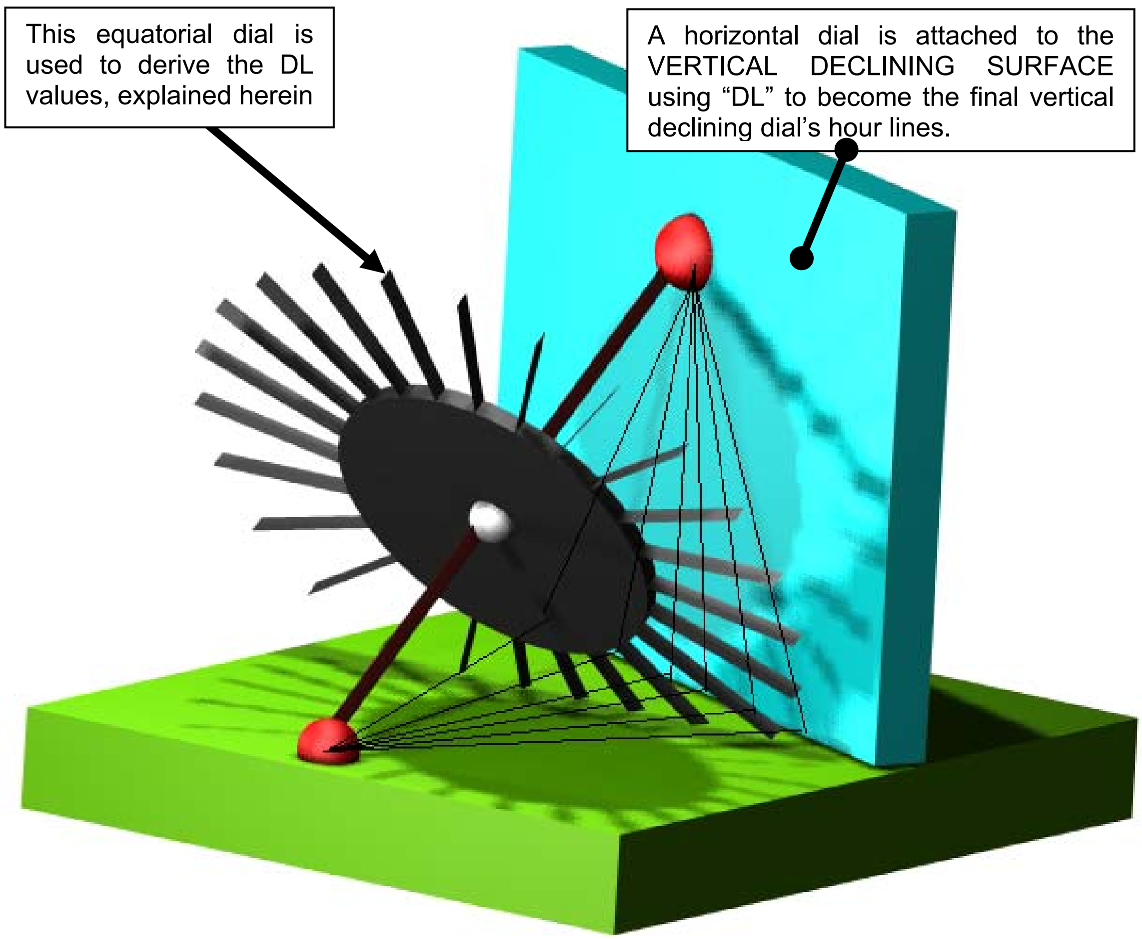
Below is the plant Earth, there is a horizontal dial at the pole, a vertical (by definition) at the equator.

Imagine the movement of the dial south from the pole to the lower latitudes, and unless tilted, there would still be a 15 degrees per hour, hour line angle. Moving to lower latitudes, a horizontal dial would be parallel to the surface, and thus tilted from where it was at the pole. As lower latitudes are reached, that tilting which keeps the dial plate horizontal with the planet's surface, causes the hour lines to depart from 15 degrees per hour, As that happens, the old housewife's trick no longer works.



VERTICAL DECLINER PROOF ~ 1

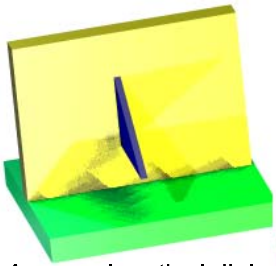
THE VERTICAL DECLINER ~ USING AN EQUATORIAL DIAL TO DERIVE “DL” ENABLING A HORIZONTAL DIAL TO BE PLACED ON THE VERTICAL SURFACE BECOMING THE VERTICAL DECLINING DIAL.



An alternative proof for the vertical declining dial which uses SD, SH, and DL. In using SH it thus uses the 15 degree radials of an equatorial dial forming a horizontal dial (yes) on that vertical declining surface, as opposed to the method used in article 2 which uses a surrogate horizontal dial on a horizontal surface. NOTE: The equatorial radial directly used is for the “DL” figure, and the other equatorial dial radials are used indirectly in the form of a horizontal (yes, not vertical) dial slapped onto the vertical declining surface or wall.

THE VERTICAL DECLINER
USING AN EQUATORIAL DIAL

The vertical decliner ~ using an equatorial dial to derive "DL" enabling a horizontal dial (yes) to be placed on the vertical surface becoming the new vertical declining dial.



A normal vertical dial

The vertical dial has been rotated "declined" but the gnomon has stayed where it was. This will work. But the gnomon's plane is not perpendicular to the dial plate.

A new gnomon has been built such that is perpendicular to the dial plate, but in so doing it has rotated. To the right, the old gnomon has been removed.

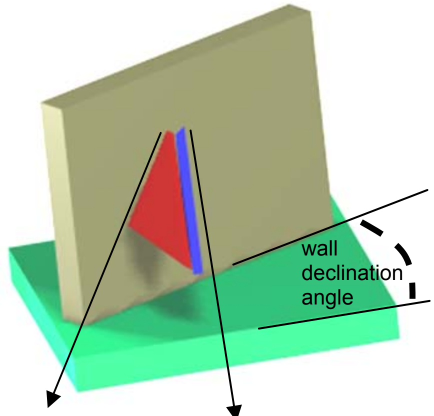
In the first picture above left, a normal vertical dial is shown. In the second picture, above middle, the dial plate has been declined (rotated about the vertical axis). In this case the gnomon has been left as it was, and this works. It is the method suggested by Winthrop Dolan in his book "A choice of Sundials". The benefit is the sub-style is still vertical and the co-latitude angle is retained in the gnomon. The drawback comes when declination curves are to be added to depict calendar information, or analemmas, or Italian or Babylonian lines.

The third picture above right is the same as the second picture, but the gnomon has been rotated so the sub-style is no longer vertical. The gnomon now looks like a normal gnomon on a normal dial plate, all of which has been rotated. The amount of rotation is called the style distance (SD), and the new angle between the style and sub-style is no longer the co-latitude but a derived angle called the style height (SH). The drawback is that SD and SH must be calculated, the benefit is that declination curves are now simple to draw as they are symmetrical around the now rotated sub-style line when extended, and the latitude they would use is now SH, and not the dial's actual latitude. And this in turn simplifies analemmas, Italian and Babylonian lines. Calculating the hour line angles in reference to the extended sub-style involves a value called the difference-in-longitude or DL. This longitude difference is nothing more than the distance between the extended sub-style line, and the vertical line that was the sub-style on the second picture above.

The fourth picture is the same as the third picture except that the original gnomon has been removed.

There are three values that we need to derive, and give those three, then we can derive the formula for a vertical declining dial based solely on a surrogate equatorial dial.

FIRST: We need to find out what it takes to rotate a gnomon from the vertical to a value that is like a standard gnomon on a normal horizontal or vertical (but non declining) dial. This would be the case in the third figure on the preceding page, reproduced here.

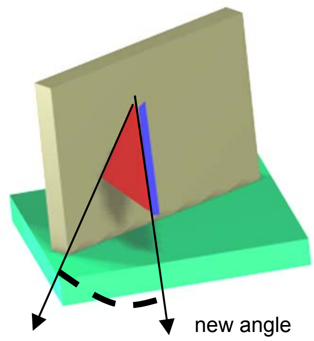


The old sub-style was vertical of the gnomon that was never "adjusted". The new adjusted gnomon has a new sub-style. The angle of rotation on the dial plate is called the style distance or SD.

As we will see, this SD angle is derived in a few pages and is:-

SD = atan (sin(wall declination angle) / tan(dial latitude))

SECOND: As a result of adjusting the gnomon, the angle that was the co-latitude of the un adjusted dial (co-latitude for vertical non declining dials, or latitude for horizontal dials) now changes. That new angle is called the style height or SH.



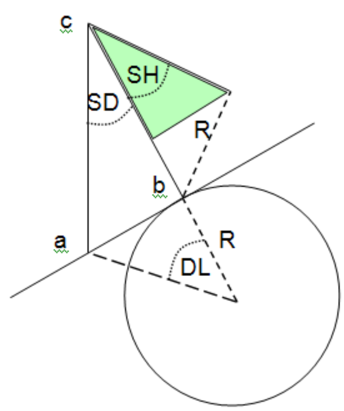
That new angle is the angle between the old gnomon style, and the dial plate, or, new sub-style.

This new angle is a pseudo latitude as we will see.

And, as we will see, this SH angle is derived in a few pages and is:-

SH = asin (cos (wall declination angle) * cos(dial latitude))

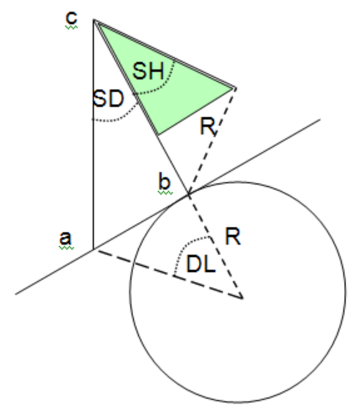
THIRD: By doing the first two steps above, we now have a normal albeit adjusted gnomon, and it has a style height (SH) and we can use that new gnomon to place an equatorial dial on its tip (nodus) tilted by the true co-latitude of the dial.



Now, in a non longitude adjusted vertical dial, local apparent time noon is a vertical line.

If we could find out what angle that surrogate equatorial dial's radial would impact that vertical noon line, we would be in business.

continued: That angle is called DL or difference-in-longitude, and is simply a figure showing how much the adjusted gnomon's vertical dial would be offset were we to design a horizontal dial to plop on that vertical dial plate.



And of course, we would add to DL the real offset of the entire dial which is the dial's longitude from the reference legal meridian.

And, as we will see, this SH angle is derived in a few pages and is:-

DL = atan (tan(SD) / sin(SH))

or with some substitutions

DL = atan (tan(wall declination angle) / (sin(dial latitude)))

FINALLY: We can now place a surrogate equatorial dial on the gnomon's style, it is tilted at true co-latitude. The style is still paralleling the earth's north south polar axis. And using DL on that equatorial dial, we have a baseline for noon, and then we offset radials every 15 degrees. And the math for that is the simple math for a normal horizontal dial of latitude SH, or a vertical dial of co-latitude SH.

So we would use:-

hla = atan(sin(SH) * tan (hour angle based on DL)

where "hla" is the hour line angle on the vertical declining dial.

And that is the point of this article, and is derived in the next few pages.

We will pay special attention to angles that are approaching 90 degrees on that surrogate equatorial dial, because tan(90) is infinity, in other words, errors may creep in with the derived formula if we are not careful. We will also pay attention to angles that do not make sense eve in the conventional formulae.

The following spreadsheet uses latitude 33.5, longitude 112.1, legal meridian of 105 (Phoenix, AZ) and a wall declination of S 10 E (-10) in the example. It shows hour line angles for 0500 and 0600 that are obviously erroneous, and this uses the formulae using a surrogate horizontal dial. The 1900 hour line angle is also erroneous.

hour line angles from vertical & substyle				
LATITUDE:	33.5	LNG: 112.1	REF:	105.0
		Corr: -7.1		
Declination:	-10	SW is +ve, SE is -ve		
TIME	Faces SE			
hh.hh	fr.vert			
5.00	70.0			
6.00	88.2		+ve angles	
7.00	-74.4		-ve angles	
8.00	-58.5		If conditional format	
9.00	-44.0		is supported	
10.00	-30.8			
11.00	-18.3			
12.00	-5.9			
13.00	6.8			
14.00	20.5			
15.00	35.5			
16.00	52.1			
17.00	70.0			
18.00	88.2			
19.00	-74.4			
STYLE:SD	-14.7	SD	from vertical	
STYLE:SH	55.2	SH	style to dial plate	34.8 90-SH
diff.lng	-10.6	DL		

The spreadsheet below uses the standard formula most texts use, and show the same issues.

TIME	TIME	DEC SxxW
hh.hh	hh.hh	-10
6.00	5.53	88.2
7.00	6.53	-74.4
8.00	7.53	-58.5
9.00	8.53	-44.0
10.00	9.53	-30.8
11.00	10.53	-18.3
12.00	11.53	-5.9
17.00	16.53	70.0
18.00	17.53	88.2
19.00	18.53	-74.4

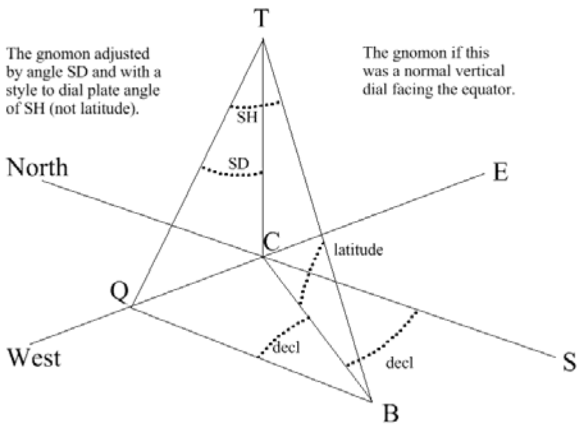
Both formulae show 88.2 degrees for 0600 hrs in the example 33.5 latitude dial, longitude 112.1 and legal meridian of 105, with a wall declination of -10 degrees, or, S10E. The angle has problems because the sign differs from its neighbor.

Similarly 1900 shows an angle of 74.4 with a sign not matching the adjacent sign.

We will now derive the formula using the surrogate equatorial dial, after deriving SD, SH, and DL.

DERIVATION OF SD (style distance) and SH (style height)

In the proof and pictorial that follows, please note where the right angle is. Also, in the math that follows, "dec" (wall declination) is the normally defined wall declination using the SxxxW terminology.



TCB is a vertical gnomon whose style is latitude and north/south, whose sub-style is vertical.

TQB is a rotated gnomon whose **style** is latitude and north/south but the style to sub-style angle is SH (style height).

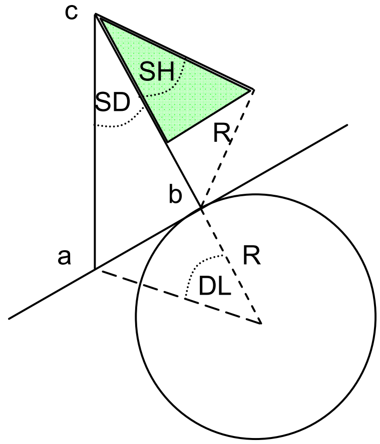
[SD]	tan (SD)	=	CQ / TC	Style Distance
thus	SD = style distance = atan (CQ / TC)			
and	tan (lat) = TC / CB	thus	TC = CB * tan (lat)	
and	sin (dec) = CQ / CB	thus	CQ = CB * sin (dec)	
and given	SD = style distance = atan (CQ / TC)			
then	SD = atan (CB * sin(dec) / CB * tan (lat))			
so				

SD = atan (sin(dec) / tan(lat))

[SH]	sin (SH)	=	QB / TB	Style Height
thus	SH = style height = asin (QB / TB)			
and	cos (lat) = CB / TB	thus	TB = CB / cos(lat)	
and	cos (dec) = QB / CB	thus	QB = CB * cos (dec)	
and given	SH = style height = asin (QB / TB)			
then	SH = asin (CB * cos (dec) / (CB / cos(lat)))			
so				

SH = asin (cos (dec) * cos(lat))

DERIVATION OF DL (distance in longitude)



"DL" is the "Difference in Longitude" and is the degrees by which local apparent noon is displaced from the perpendicular from the equatorial dial to the equinox line. In essence, this is the longitude at which a non-longitude corrected h-dial would match this resulting vertical decliner.

sin(SH) = R / cb	thus	cb = R / sin(SH)	
tan(SD) = ab / cb	thus	tan(SD) = ab * sin(SH) / R	and
ab = cb * tan(SD)	and	ab = tan(SD) * R / sin(SH)	also
tan(DL) = ab / R	thus	DL = atan(ab / R) = atan(ab * 1/ R)	
	and	DL = atan((R * tan(SD) / sin(SH)) * 1/R)	
	so		

DL = atan (tan(SD) / sin(SH))

being the radial for Local apparent noon

but SD = atan(sin(dec) / tan(lat)) from earlier
and SH = asin (cos (dec) * cos(lat)) from earlier

so since DL = atan (tan(atan(sin(dec) / tan(lat))) / sin(SH))

hence DL = atan ((sin(dec)*cot(lat)) / sin(SH)) [cot = 1/tan]

then DL = atan ((sin(dec)*cot(lat)) / sin(asin (cos (dec) * cos(lat))))

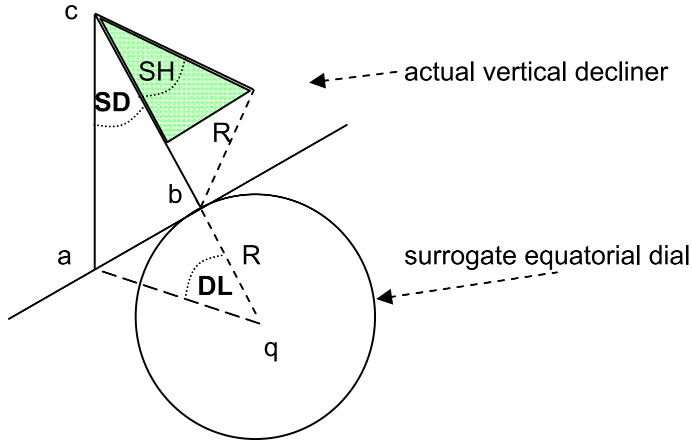
hence DL = atan ((sin(dec)*cot(lat)) / (cos (dec) * cos(lat)))

$$DL = \text{atan} \left(\frac{(\sin(dec) * \cot(lat))}{(\cos(dec) * \cos(lat))} \right)$$

DL = atan (tan(dec) / (sin(lat)*tan(lat)/tan(lat))) [cos = sin/tan]

DL = atan (tan(dec) / (sin(lat)))

DERIVATION OF "hla" (hour line angle) Now we have:



SD	=	atan (sin(dec) / tan(lat))
SH	=	asin (cos(dec) * cos(lat))
DL	=	atan(tan(sd) / sin(sh))
		or if we add in the dial's longitude and legal meridian then
DL	=	atan(tan(sd) / sin(sh)) + (long – ref) or...
DL	=	atan (tan(dec) / sin(lat)) + (long – ref)

And now we need an hour line angle from "**ca**" for an hour angle on the surrogate equatorial dial. The theory basis is to use "ca" as noon LAT, and move hour lines either side using a latitude of SH.

We could then look at the surrogate equatorial dial and run 15 degrees back from line "qa". However, it might be simpler to use the horizontal (yes, horizontal not vertical) dial formula which is based on the equatorial dial, and use a latitude of SH. This would require identifying the angle on the equatorial dial that generates a noon line of "ca", and we have that angle, it is "DL".

For a horizontal dial we know that the hour line angle "hla" the hour line angle is:-

hla = atan(sin(lat) * tan (ha)) or in this case...
hla = atan(sin(SH) * tan (ha))

where "ha" is the hour angle on the surrogate equatorial dial, which is:-

ha = (hour from noon * 15) + DL

thus the hour line angle for the vertical decliner is:-

hla = atan(sin(SH) * tan((hour from noon * 15) + DL))
= atan(sin(asin(cos(dec)*cos(lat))) * tan((hour from noon * 15) + DL))

= **atan (**
cos(dec) * cos(lat) * tan((hr*15)+atan(tan(dec)/sin(lat)) + (long–ref))
)

or in spreadsheet form (Excel) where the cells are named "latitude", "decl", "lng" and "ref" and where "k8" if the hour, actually, k8, k9, ... k.whatever

```
=DEGREES(ATAN(COS(RADIANS(latitude))*COS(RADIANS(decl)) * TAN(RADIANS(
        (15*(12-K8))+DEGREES(ATAN(TAN(RADIANS(decl))/SIN(RADIANS(latitude))))+(lng-ref)))))
```

The vertical decliner hour line angles based on a surrogate equatorial dial and then a horizontal dial (yes) on the vertical surface

This was implemented in a spreadsheet also (lat=33.5, lng=112.1, ref=105, dec=-10) :-

LIMIT TO HA LESS THAN:		90			hla from SD		
Angles on the "equatorial" dial							
hr	*15	+dl		minus **SD**	hla		
5.00	105	94.38290955	ha>limit	-70.0	-84.7		
6.00	90	79.38290955		91.8	77.1	???	???
7.00	75	64.38290955		74.4	59.7		
8.00	60	49.38290955		58.5	43.8		
9.00	45	34.38290955		44.0	29.3		
10.00	30	19.38290955		30.8	16.1		
11.00	15	4.382909554		18.3	3.6		
12.00	0	-10.6170904		5.9	-8.8		
13.00	-15	-25.6170904		-6.8	-21.5		
14.00	-30	-40.6170904		-20.5	-35.2		
15.00	-45	-55.6170904		-35.5	-50.2		
16.00	-60	-70.6170904		-52.1	-66.8		
17.00	-75	-85.6170904		-70.0	-84.7		
18.00	-90	-100.61709	ha>limit	91.8	77.1	???	???
19.00	-105	-115.61709	ha>limit	74.4	59.7		
				hla from vertical			

hh.hh	fr.vert	from SD
5.00	70.0	84.7
6.00	88.2	102.9
7.00	-74.4	-59.7
8.00	-58.5	-43.8
9.00	-44.0	-29.3
10.00	-30.8	-16.1
11.00	-18.3	-3.6
12.00	-5.9	8.8
13.00	6.8	21.5
14.00	20.5	35.2
15.00	35.5	50.2
16.00	52.1	66.8
17.00	70.0	84.7
18.00	88.2	102.9
19.00	-74.4	-59.7

The formula derived in this article

conventional formulae

The “ha>limit” was an arbitrary limit set at an hour angle on the equatorial dial of 90 degrees in this case.

The “???” says that the angle in the two respective columns differs from the conventional formulae. Comparing them, in each case it will be seen that the conventional formula also produces an anomalous hour line angle. Review shows it to be a change in the basis for the angle in question. For more analysis, see “Nagging questions” immediately following.

Conclusion: The following formulae have been derived and are usable.

SD = atan (sin(dec) / tan(lat))
SH = asin (cos(dec) * cos(lat))

DL = atan(tan(sd) / sin(sh))
or if we add in the dial’s longitude and legal meridian then
DL = atan(tan(sd) / sin(sh)) + (long – ref) or...
DL = atan (tan(dec) / sin(lat)) + (long – ref)

and hla = atan(cos(dec) * cos(lat) * tan((hr*15) + atan(tan(dec)/sin(lat)) + (long–ref)))

This is the hour line angle from the vertical, useful in drafting dial plates. To convert to the angle from the sub-style, simple subtract the SD (style distance)

hla.fr.sd= hla – SD

The spreadsheet associated with this article is: ~ XLS bk3 sup Vdec angles using Q dial.xls

NAGGING QUESTIONS YOU MAY HAVE

QUESTION: So why do the formulae differ at 0600 and 1800?

ANSWER: A closer look shows that the error as such is not a numerical error, it is where the basis for that angle changed. The two columns “=180-left” show this to be the case.

SUPPLEMENT ARTICLE 1 FORMULA

LIMIT TO "HA" LESS THAN: 88

surrogate dial angles						hla from sub-style			
hr	*15	+dl		minus SD	=180-left	basic hla	=180-left		
5.00	105	94.383	ha>limit	-70.0	250.0	-84.7	264.7		
6.00	90	79.383	ha>limit	91.8	88.2	77.1	102.9	???	???
7.00	75	64.383		74.4	105.6	59.7	120.3		
8.00	60	49.383		58.5	121.5	43.8	136.2		
9.00	45	34.383		44.0	136.0	29.3	150.7		
10.00	30	19.383		30.8	149.2	16.1	163.9		
11.00	15	4.383		18.3	161.7	3.6	176.4		
12.00	0	-10.617		5.9	174.1	-8.8	188.8		
13.00	-15	-25.617		-6.8	186.8	-21.5	201.5		
14.00	-30	-40.617		-20.5	200.5	-35.2	215.2		
15.00	-45	-55.617		-35.5	215.5	-50.2	230.2		
16.00	-60	-70.617		-52.1	232.1	-66.8	246.8		
17.00	-75	-85.617		-70.0	250.0	-84.7	264.7		
18.00	-90	-100.617	ha>limit	91.8	88.2	77.1	102.9	???	???
19.00	-105	-115.617	ha>limit	74.4	105.6	59.7	120.3		
				hour line angle from vertical					

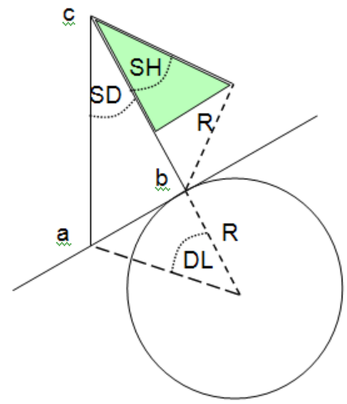
In all cases, the mismatch is because the hour angle exceeds 90 degrees or the conventional formula also has a problem.

For example, look at the “correct” values from the conventional formula, they are:-

6.00	88.2	compared to this alternative formula:-	91.8	88.2

The conventional formula returns 88.2 degrees, the alternative formula returns 91.2, however when taken from 180 (a switch in the base for the angle), the result is 88.2. Other mismatches have similar explanations.

QUESTION: Surely the picture to the right, which is used for formula derivation, has the wrong orientation for wall's declination. It shows an SD suggesting a SW facing plate yet the line "ab" shows a line facing SE.



ANSWER: The geometry shown is merely SD and SH with DL being derived. There is no showing of the wall's declination. The line "ab" is not the orientation of the wall, it is a line perpendicular to the sub style line "CB".

QUESTION: This is a vertical decliner, so why is a "horizontal" dial used in the hour line angle derivation, and not a vertical dial.

ANSWER: A vertical dial could be used, but the formula is:-

hla = atan (tan (lha) * cos (lat))   and "lat" is then "90-SH"

and since

cos(90-SH) is sin(SH)

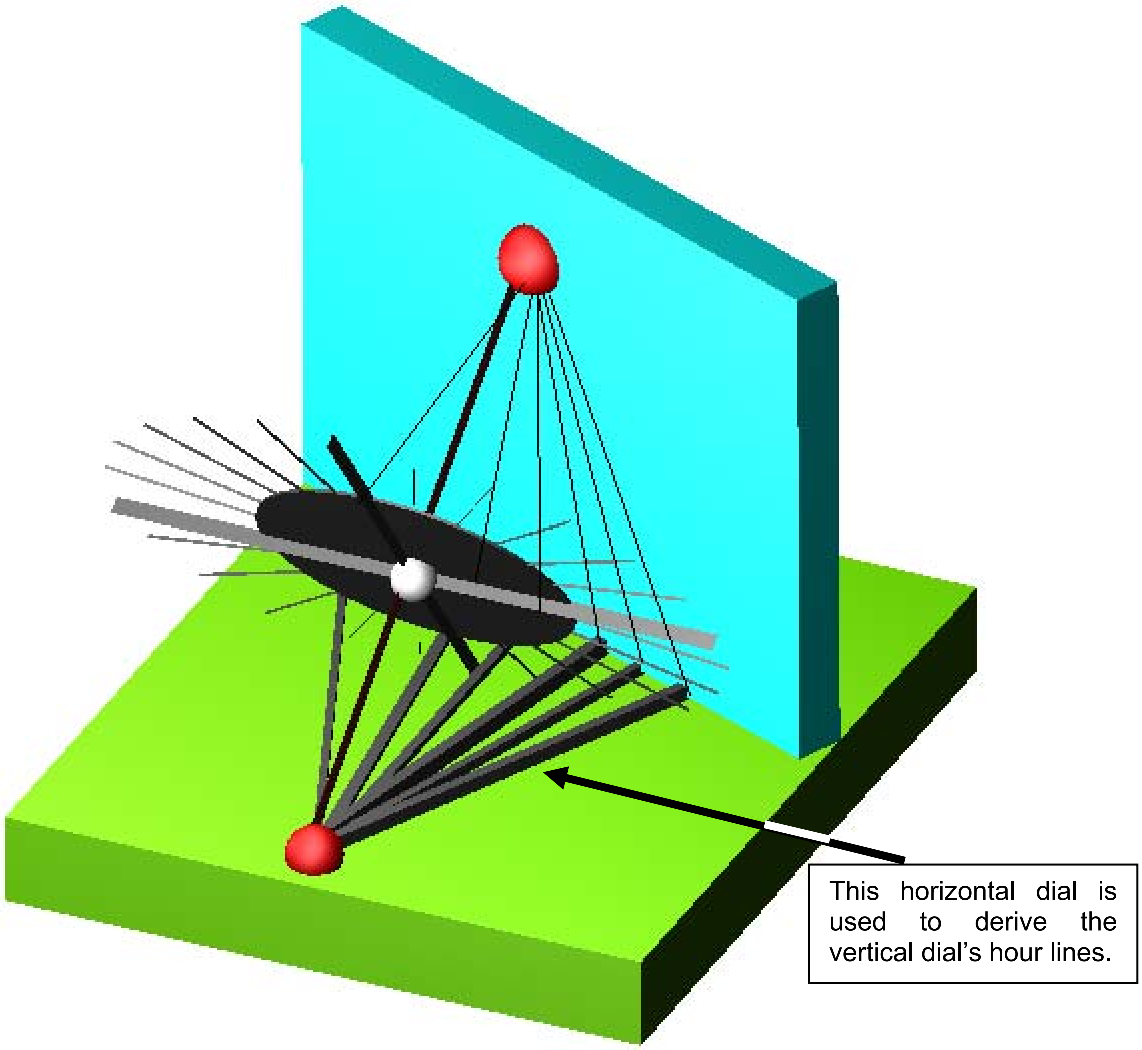
the result is the standard horizontal dial formula.

QUESTION: Why not continue to use equatorial dial radials such as was used when deriving "DL", seems simple enough.

ANSWER: It may seem simple however the projective geometry actually becomes quite complex because the distance from the nodus to the dial plate, while correct for the "DL" presentation, is not simple because the distance has to be derived from the third dimension, whereas the "DL" presentation is a two dimensional projection. That is why the equatorial dial was used for deriving "DL", but not used for the vertical decliner hour line angles, when we switched to a surrogate horizontal dial slapped onto the vertical declining wall or surface. This technique was also used in declining dial declination curves that depict calendar information, and for vertical declining dials with analemmas depicted.

VERTICAL DECLINER PROOF ~ 2

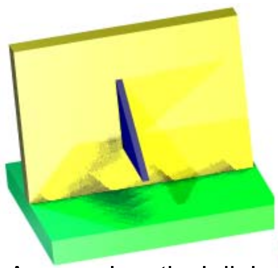
THE VERTICAL DECLINER
USING A HORIZONTAL DIAL ON A HORIZONTAL SURFACE



An alternative proof for the vertical declining dial which uses a surrogate horizontal dial. Whereas article 1 used a horizontal dial placed on the vertical declining wall or surface, this uses the horizontal dial where horizontal dials go, namely on a horizontal surface. This proof is also contained in the main appendices, but repeated here for consistency.

THE VERTICAL DECLINER
USING A HORIZONTAL DIAL

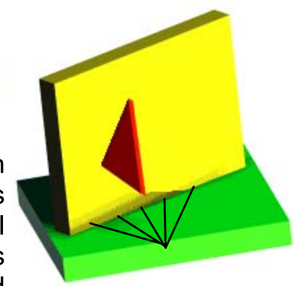
An alternative proof for the vertical declining dial which uses a surrogate horizontal dial in a horizontal position as opposed to article 1 which used one superimposed on the vertical declining dial plate, surface, or wall..



A normal vertical dial

The vertical dial has been rotated “declined” but the gnomon has stayed where it was. This will work. But the gnomon’s plane is not perpendicular to the dial plate.

A new gnomon has been built such that is perpendicular to the dial plate, but in so doing it has rotated. To the right, the old gnomon has been removed.

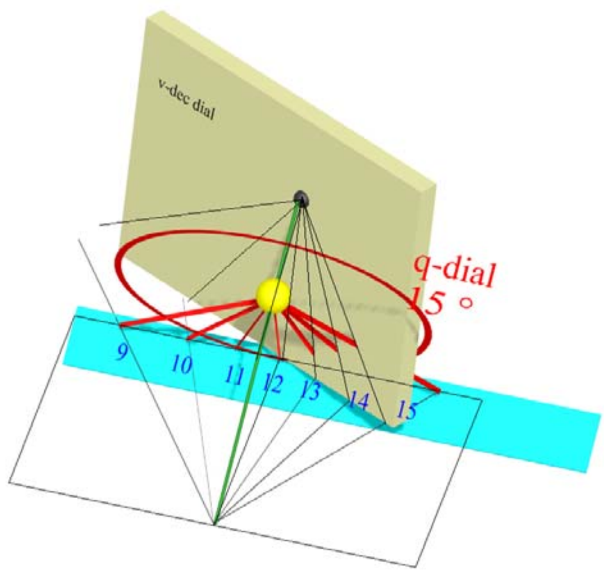


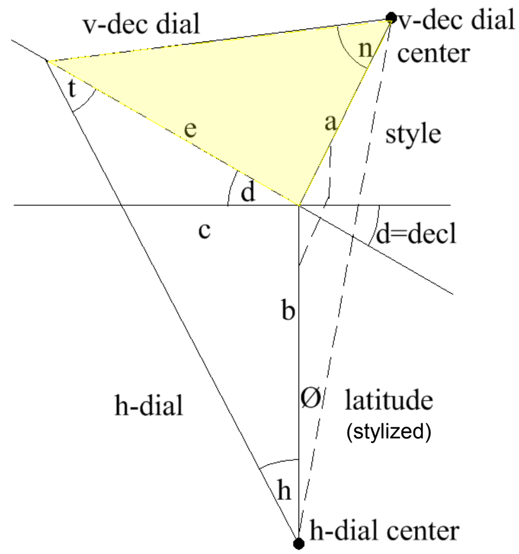
In the first picture above left, a normal vertical dial is shown. In the second picture, above middle, the dial plate has been declined (rotated about the vertical axis). In this case the gnomon has been left as it was, and this works. It is the method suggested by Winthrop Dolan in his book “A choice of Sundials”. The benefit is the sub-style is still vertical and the co-latitude angle is retained in the gnomon. The drawback comes when declination curves are to be added to depict calendar information, or analemmas, or Italian or Babylonian lines.

The third picture above right is the same as the second picture, but the gnomon has been rotated so the sub-style is no longer vertical. The gnomon now looks like a normal gnomon on a normal dial plate, all of which has been rotated. The amount of rotation is called the style distance (SD), and the new angle between the style and sub-style is no longer the co-latitude but a derived angle called the style height (SH). The drawback is that SD and SH must be calculated, the benefit is that declination curves are now simple to draw as they are symmetrical around the now rotated sub-style line when extended, and the latitude they would use is now SH, and not the dial’s actual latitude. And this in turn simplifies analemmas, Italian and Babylonian lines. Calculating the hour line angles in reference to the extended sub-style involves a value called the difference-in-longitude or DL. This longitude difference is nothing more than the distance between the extended sub-style line, and the vertical line that was the sub-style on the second picture above.

Proof of Decliner/Great Decliner Hour Line angles

In the stylized figure to the right below, the triangle "h-dial" has sides "b", and "c" and angle "h". Side "b" is the horizontal dial's sub-style and a selected horizontal dial's hour line angle "h". Triangle "v-dec dial" has two named sides, with angle "n" being the vertical decliner's equivalent hour line angle that is associated with the horizontal dial's angle "h". Both dials share a style that connects the h-dial and v-dec dial centers, shown by a depicted dashed line. The vertical decliner's sub-style is not depicted in the figures below, and their "SD" and "SH" (style angular distance and angular height) are derived elsewhere. Declination is "d" and "Ø" is latitude.





$$a = b * \tan(Ø) \qquad [1]$$

$$\tan(n) = e / a \quad \text{thus ...} \quad n = \operatorname{atan}(e / a) \qquad [2] \text{ desired}$$

so $$n = \operatorname{atan}(e / b * \tan(Ø)) \qquad [3]$$

$$e / \sin(h) = b / \sin(180 - (h + (90 + d))) \qquad [4] \text{ law of sines}$$

so $$e / \sin(h) = b / \sin(90 - h - d)$$

thus $$e = b * \sin(h) / \sin(90 - h - d) \qquad [5]$$

$$n = \operatorname{atan}(e / a) = \operatorname{atan}\left[\frac{b * \sin(h) / \sin(90 - h - d)}{b * \tan(Ø)}\right] \qquad [\text{using } 2, 5, 1]$$

so $$n = \operatorname{atan}\left[\frac{b * \sin(h)}{\sin(90 - h - d) * b * \tan(Ø)}\right]$$

thus $$n = \operatorname{atan}\left[\frac{\sin(h)}{\sin(90 - h - d) * \tan(Ø)}\right] \qquad [6]$$

and using

h = atan (sin (Ø) * tan (sun hour angle)) [for h-dial]

and since from the prior page

n = atan (sin (h) / tan (Ø) * sin(90 – h – d)) [from 6]

then using the sun's hour angle as opposed to a surrogate horizontal dial's hour line angles

then

$$n = \text{atan}\left[\frac{\text{sin}(\ \text{atan}(\ \text{sin}(\ Ø\ ) * \text{tan (sun hour angle)}\ )\ )}{\text{tan}(\ Ø\ ) * \text{sin}(\ 90 - d - \text{atan}(\text{sin}(\ Ø\ ) * \text{tan (sun hour angle)})\ )}\right]$$

[7]

Hence, considering a spreadsheet or a procedural program implementation of a vertical dial that declines, it has hour line angles "n" equal to:-

i.e. n = DEGREES(ATAN(SIN(RADIANS((DEGREES(ATAN(TAN

(RADIANS(15*(12-hr)+d.long))*SIN(RADIANS(lat))))))) /

(TAN(RADIANS(lat))*SIN(RADIANS(90-dec-

(DEGREES(ATAN(TAN(RADIANS(15*(12-hr)+d.long))*SIN(RADIANS(lat))

))))))))

where the hour itself, and longitude corrections are all considered.

NOTE: The formula derived above can be converted to the simpler form using methods beyond the scope of the trigonometry discussed in this book. However, the formula is presented as is in order to show the derivation in a logical and simple form.

The results of the formula above match the formula usually published which is:-

n = **atan (cos(Ø) / (cos(dec) cot(ha) + sin(dec) sin(Ø))**

The Illustrating Shadows formula derived above is used in one of the vertical decliner sheets in:-

illustratingShadows.xls

and the standard common formula is also used in another vertical decliner sheet. The index of sheets as well as the individual sheets themselves clearly state which formula is used.

This was implemented in a spreadsheet also (lat=33.5, lng=112.1, ref=105, dec=-10) :-

hour line angles from vertical & substyle

LATITUDE:	**33.5**	**LNG:**	**112.1**	**REF:**	**105.0**
		Corr:	-7.1	Corr h.h	-0.473
Declination:	**-10**	SW is **+**ve, SE is **-**ve			

TIME hh.hh	Faces SE fr.vert	from sub style
5.00	69.97	84.67
6.00	88.16	102.86
7.00	-74.42	-59.72
8.00	-58.46	-43.76
9.00	-44.03	-29.33
10.00	-30.82	-16.12
11.00	-18.30	-3.60
12.00	-5.95	8.75
13.00	6.79	21.49
14.00	20.46	35.16
15.00	35.50	50.20
16.00	52.11	66.81
17.00	69.97	84.67
18.00	88.16	102.86
STYLE:SD	**-14.7**	**SD** from vertical
STYLE:SH	**55.2**	**SH**
diff.lng	**-10.6**	**DL**

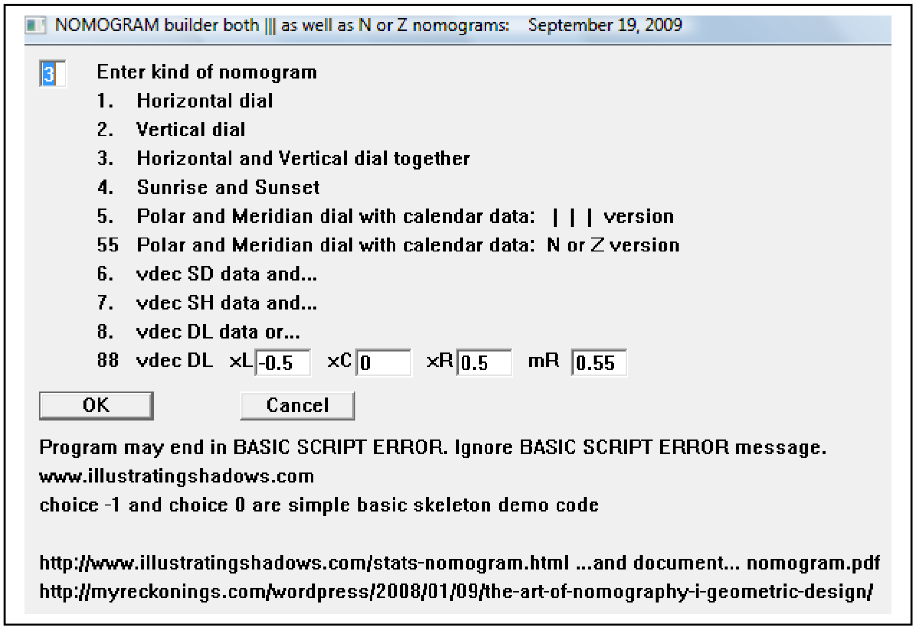
CONVENTIONAL FORMULA

time	v.dec.hr.line.angle	
5	-69.97	-84.67
6	-88.16	-102.86
7	74.42	59.72
8	58.46	43.76
9	44.03	29.33
10	30.82	16.12
11	18.30	3.60
12	5.95	-8.75
13	-6.79	-21.49
14	-20.46	-35.16
15	-35.50	-50.20
16	-52.11	-66.81
17	-69.97	-84.67
18	-88.16	-102.86
19	74.42	59.72
hour		hr.ln from SD sub style

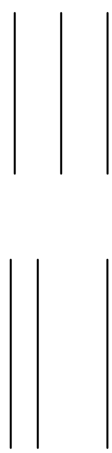
THIS DEVELOPED FORMULA

DESIGNING NOMOGRAMS FOR SUNDIALS

How nomograms work, how to design them, and specific examples of code for their creation. Illustrating Shadows provides DeltaCAD programs to generate nomograms, as well as a Pascal (Lazarus) stand alone .EXE program, and you can manipulate the parameters to see the results when values change.



Type 1



C = L + R

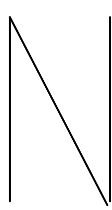
works for multiplication if you use logs

log C = log L + log R

i.e.

logs enable
C = L * R

Type 2



C = L / R

with no logs

THREE PARALLEL LINES

(Type 1)

$$C = L + R$$

works for

$$\log C = \log L + \log R$$

and thus
if you use logs it works for

$$C = L * R$$
and
$$C = L / R$$

NOTE:

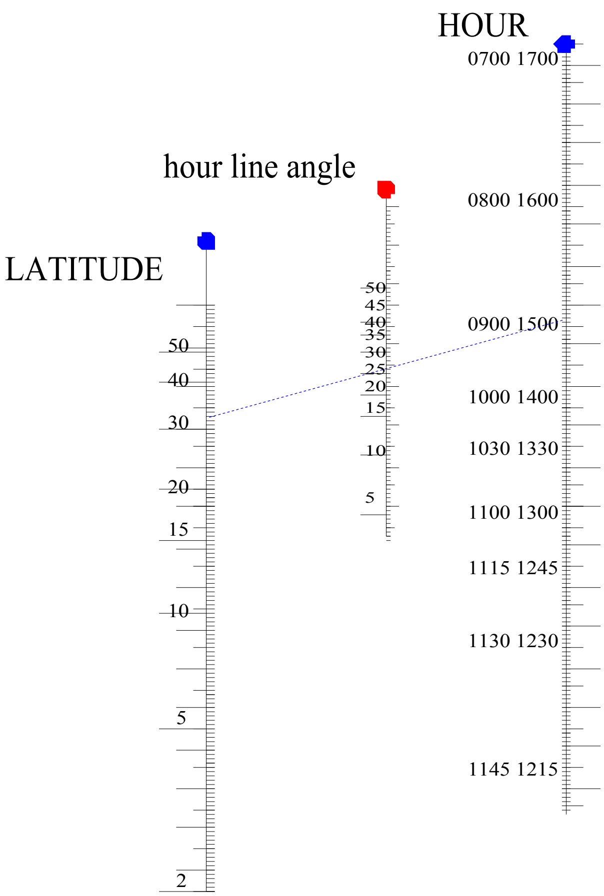
Chapter 31 of Illustrating Time's Shadow expands upon this topic.

The nomogram and the sun dial

Nomograms are graphical solutions to mathematical problems, and in the case of the horizontal dial, we have two variables (latitude and solar hour angle) and one solution (hour line angle).

hour line angle = atan (sin(lat) * tan(hour angle))

The nomogram, popularized in 1880 by Philbert d'Ocagne, has many forms, the simplest is two variables represented by vertical lines left and right, with the solution represented by a vertical line mid way between the two. There are a number of web sites that discuss nomogram theory and design, and some earlier books are also available for online study. The JAVA alternative Python has software for drafting nomograms, and at least one online JAVA tool exists for nomograms as well. In the example below, the horizontal dial nomogram was derived from scratch. In essence the nomogram is a graphical adding machine. Because the horizontal dial formula uses multiplication, logarithms are used which allow multiplication by an additive process. This way adding still works, while solving a multiplication problem.



The left scale is the logarithm of the sine of the latitude, the right is the logarithm of the tangent of the hour from noon (times 15 of course), with the center vertical line being the logarithm of the tangent of the hour line angle on the dial plate compressed by a factor of two. The axes are labeled with latitude, hour, and hour line angle for simplicity, not their logarithmic equivalents. The end result is a nomogram whereby a line can be drawn from the latitude (left) to the hour from noon (right) and the hour line angle read in the middle.

This is not longitude corrected, you add the longitude correction before using the tool.

A sample line for latitude 32, for 0900 or 1500 is shown, and the hour angle in the middle is seen to be close to 27.92 degrees which it should be.

The Hour Line Angle and the Hours From Noon scales get smaller from the bottom as you move up, but then get bigger again. This is because initially the TAN is slowly increasing but the log decreases more rapidly, however later the TAN increases far more than the log decreases.

Illustrating Shadows provides a DeltaCAD macro to draw dial nomograms. The DeltaCAD macro for nomograms is:-

nomogram.bas

This macro is interesting since logarithms base 10 did not work as documented at the time, so the logs used were base 2.718 or Napierian or natural logs.

The first step in building nomogram is to look at the extremes of the three variables because they must be placed on the nomogram's vertical lines. Their magnitudes should be reasonable and should be selected so that extremes are omitted if the nomogram scales would be impractical.

input	log sin (latitude)	–1.76 to 0
input	log tan (hour angle or time)	–1.18 to +0.57
result	log tan (hour line angle)	–1.76 to +1.06 (actually can be twice this range)

The practical latitude range for a sundial is between 0.00 to -1.76 as shown by the spreadsheet to the right. By practical, what is meant is the range of the logarithm of the sine of the latitude should fit within a reasonable scale, excessive numbers would not be employed. This is the first input.

LATITUDE	log sin(lat)	= LOG10(SIN(RADIANS(lat)))
0	#NUM!	
1	-1.76	
2	-1.46	
5	-1.06	
10	-0.76	
15	-0.59	
20	-0.47	
25	-0.37	
30	-0.30	
35	-0.24	
40	-0.19	
45	-0.15	
50	-0.12	
55	-0.09	
60	-0.06	
65	-0.04	
70	-0.03	
75	-0.02	
80	-0.01	
85	0.00	
90	0.00	

HOURS FROM NOON	log tan(hr)	=LOG10(TAN(RADIANS(15*hrs)))
0.00	#NUM!	
0.25	-1.18	
0.50	-0.88	
0.50	-0.88	
0.75	-0.70	
1.00	-0.57	
1.25	-0.47	
1.50	-0.38	
1.75	-0.31	
2.00	-0.24	
2.50	-0.12	
3.00	0.00	
3.50	0.12	
4.00	0.24	
5.00	0.57	
6.00	16.21	
7.00	#NUM!	
8.00	#NUM!	

The practical hour range from noon for a sundial is -1.18 to 0 to 0.57 as shown by the spreadsheet to the left. By practical, what is meant is the range of the logarithm of the tangent of the hours from noon should fit within a reasonable scale, excessive numbers would not be employed. This is the second input.

The practical dial plate's hour line angle range for a sundial is -1.76 to 1.06 as shown by the spreadsheet to the right. By practical, what is meant is the range of the logarithm of the tangent of the hour line angle should fit within a reasonable scale, excessive numbers would not be employed. Note that ATAN is not used, but rather TAN is, see the formula development below.

hour LINE angle		log(tan) of resulting hour line angle
		#NUM!
1		-1.76
5		-1.06
10		-0.75
15		-0.57
20		-0.44
25		-0.33
30		-0.24
35		-0.15
40	angle	-0.08
45		0.00
50		0.08
55		0.15
60		0.24
65		0.33
70		0.44
75		0.57
80		0.75
85		1.06
90		16.21

hla = atan(sin(lat) * tan(hour angle))
thus

tan(hla) = sin(lat) * tan(hour angle)
thus

log tan(hla) = log sin(lat) + log tan(hour angle)

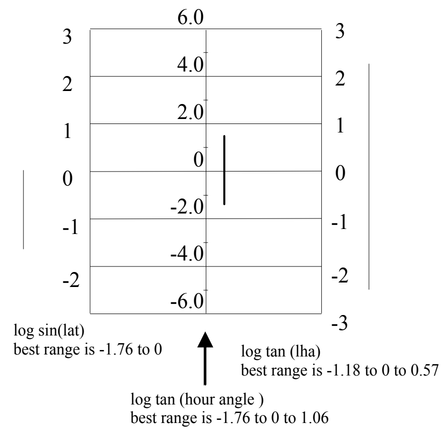
NOTE: This scale is compressed by a factor of two, see next page.

The next step is to build a set of three parallel lines, equidistant for simplicity, with linear or equally spaced number scales that cover the ranges of the inputs and the output. These ranges once again are:-

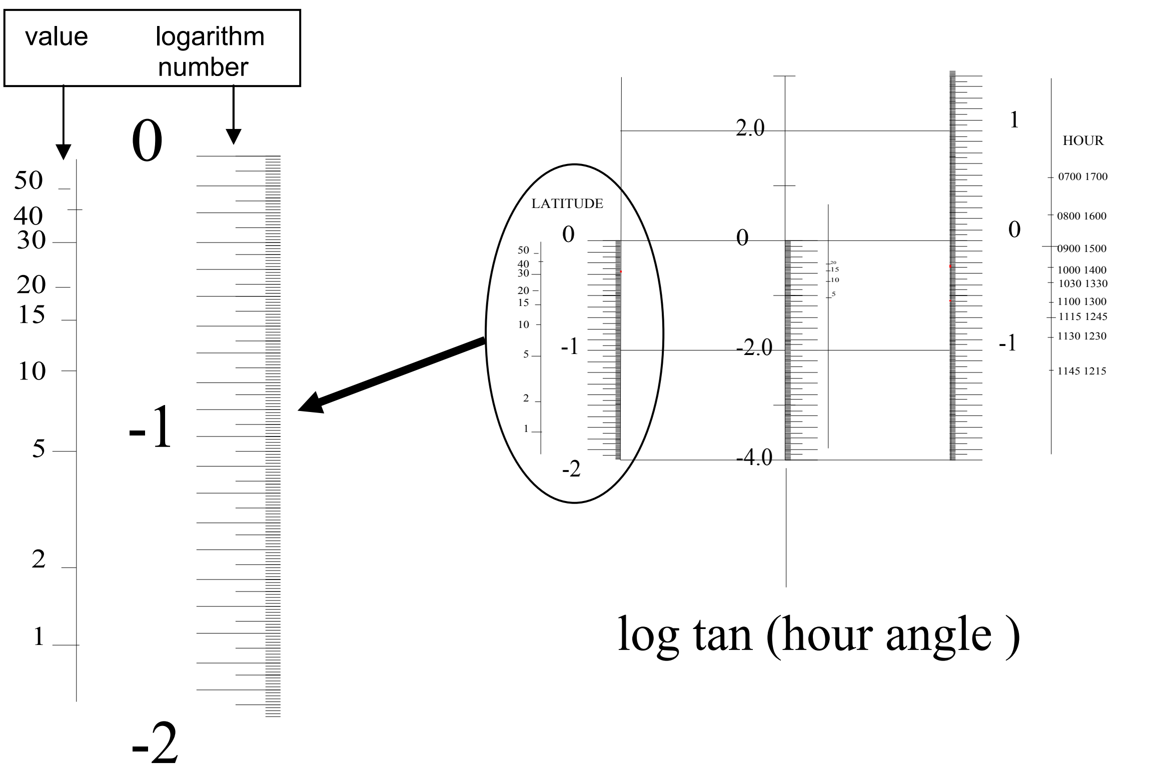
latitude range:	-1.76 to 0	log(latitude)
hour from noon range:	-1.18 to 0 to 0.57	log(hour*15)
resulting hour line angle:	-1.76 to 0 to 1.06	log(tan(hour line angle)

Thus a nomogram with three lines ranging from -1.76 to 1.06 would be appropriate.

NOTE: The center scale, when equidistant between the other two lines, has its scale compressed in half. In other words the center solution scale can accommodate numbers twice as large as can the other scales.



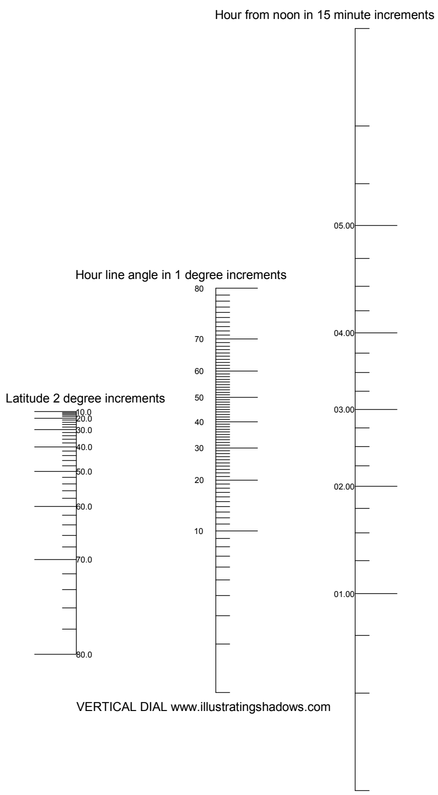
While the scale is linear, we will place the logarithm of our value at the right place, make a marker for that logarithm, and then label it with the original number (input) or final number (result), i.e.“30” for latitude 30, and not the logarithm of the sin(lat).



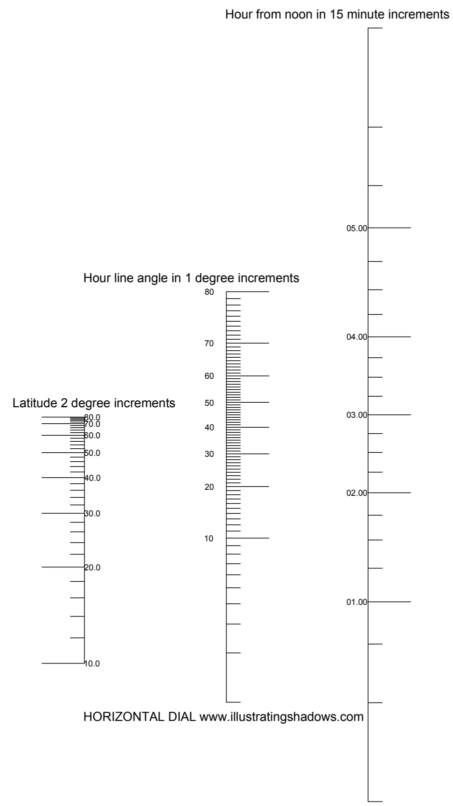
The final nomogram is shown at the start of this section, and several DeltaCAD automated nomograms are shown in the appendices.

DeltaCAD nomograms – nomogram.bas – is interesting since logarithms base 10 did not appear to work, so the logs that did were Napierian or natural logs, base 2.718, so they were used instead.

DeltaCAD nomograms – vertical dial



DeltaCAD nomograms – horizontal dial



Useful web sites:

JAVA online:

http://www.ece.rochester.edu/~jones/NomoDevel/nomogram.htm

The Art of The Nomogram - read this first

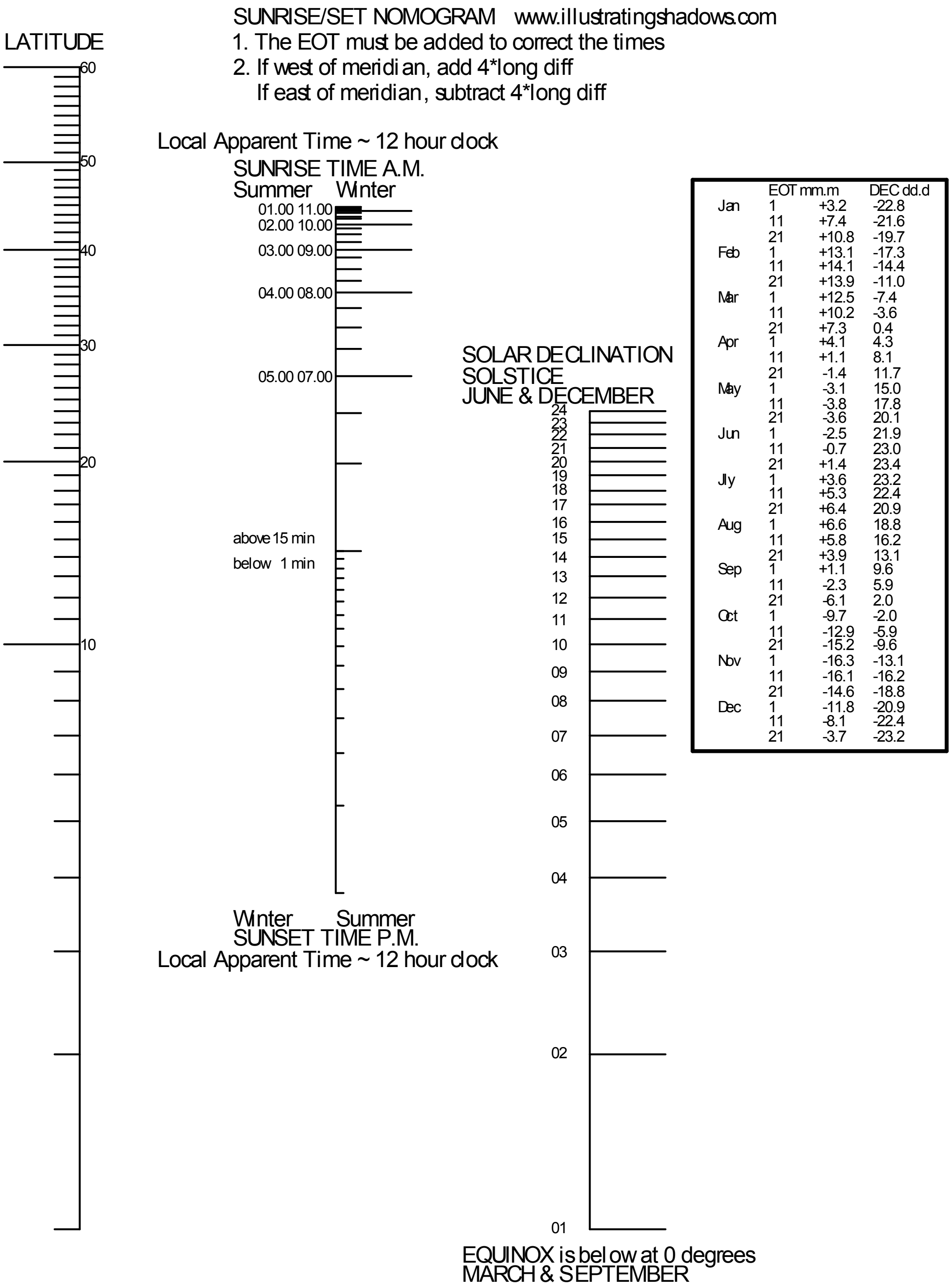
http://myreckonings.com/wordpress/2008/01/09/the-art-of-nomography-i-geometric-design/

Graphical and Mechanical Computation

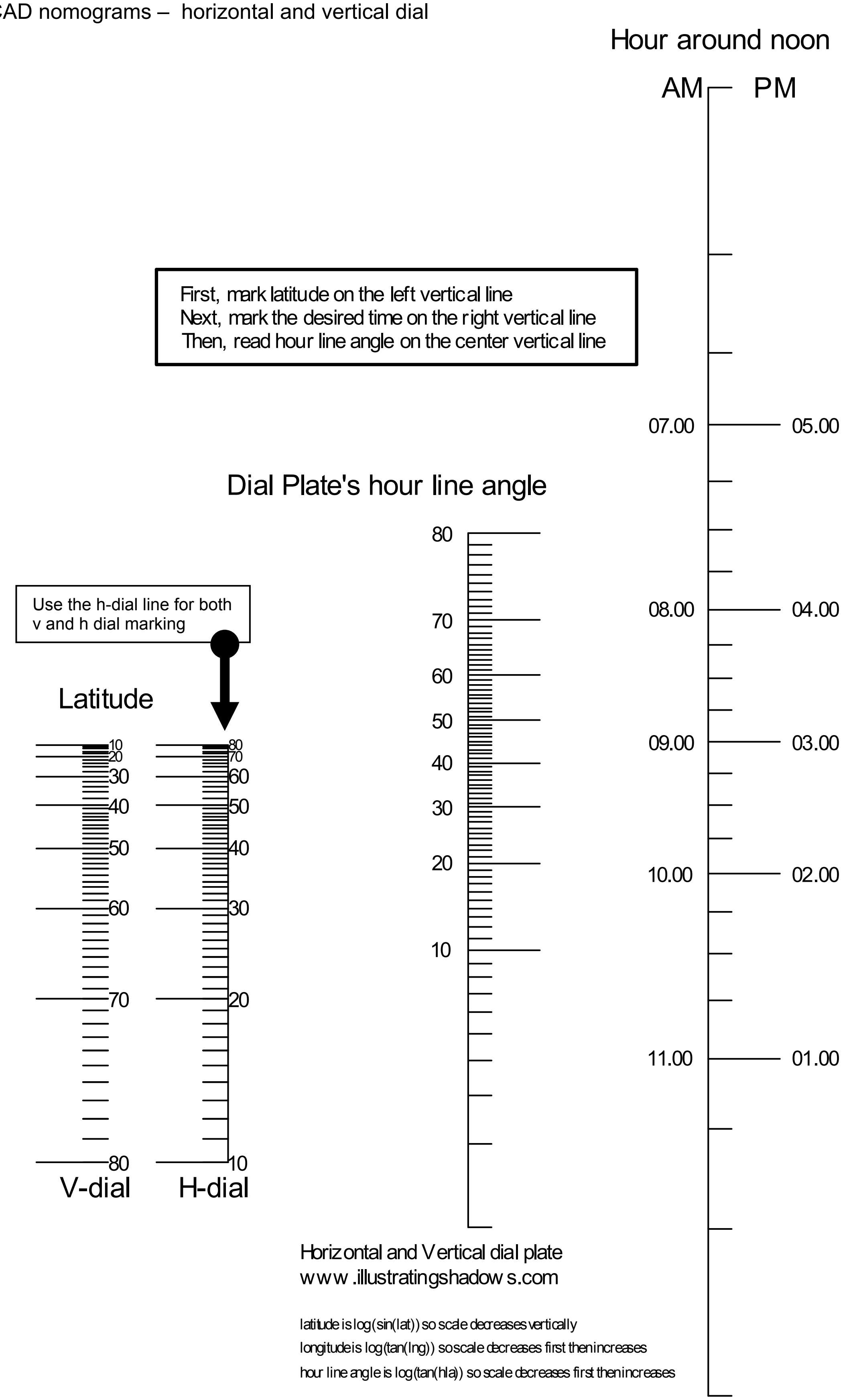
http:// www.myreckonings.com/wordpress/wp-content/uploads/Graphical_and_Mechanical_Computation.pdf

Creating nomograms with pynomo software

http:// www.myreckonings.com/wordpress/



		EOT mm.m	DEC dd.d
Jan	1	+3.2	-22.8
	11	+7.4	-21.6
	21	+10.8	-19.7
Feb	1	+13.1	-17.3
	11	+14.1	-14.4
	21	+13.9	-11.0
Mar	1	+12.5	-7.4
	11	+10.2	-3.6
	21	+7.3	0.4
Apr	1	+4.1	4.3
	11	+1.1	8.1
	21	-1.4	11.7
May	1	-3.1	15.0
	11	-3.8	17.8
	21	-3.6	20.1
Jun	1	-2.5	21.9
	11	-0.7	23.0
	21	+1.4	23.4
Jly	1	+3.6	23.2
	11	+5.3	22.4
	21	+6.4	20.9
Aug	1	+6.6	18.8
	11	+5.8	16.2
	21	+3.9	13.1
Sep	1	+1.1	9.6
	11	-2.3	5.9
	21	-6.1	2.0
Oct	1	-9.7	-2.0
	11	-12.9	-5.9
	21	-15.2	-9.6
Nov	1	-16.3	-13.1
	11	-16.1	-16.2
	21	-14.6	-18.8
Dec	1	-11.8	-20.9
	11	-8.1	-22.4
	21	-3.7	-23.2

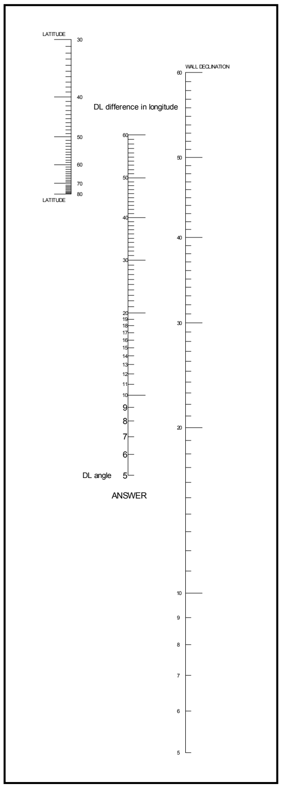

Hour around noon
AM
PM
First, mark latitude on the left vertical line
Next, mark the desired time on the right vertical line
Then, read hour line angle on the center vertical line
07.00
05.00
Dial Plate's hour line angle
80
70
60
50
40
30
20
10
08.00
04.00
09.00
03.00
10.00
02.00
11.00
01.00
Use the h-dial line for both
v and h dial marking
Latitude
10
20
30
40
50
60
70
80
V-dial
80
70
60
50
40
30
20
10
H-dial
Horizontal and Vertical dial plate
www.illustratingshadows.com
latitude is log(sin(lat)) so scale decreases vertically
longitude is log(tan(lng)) so scale decreases first then increases
hour line angle is log(tan(hla)) so scale decreases first then increases

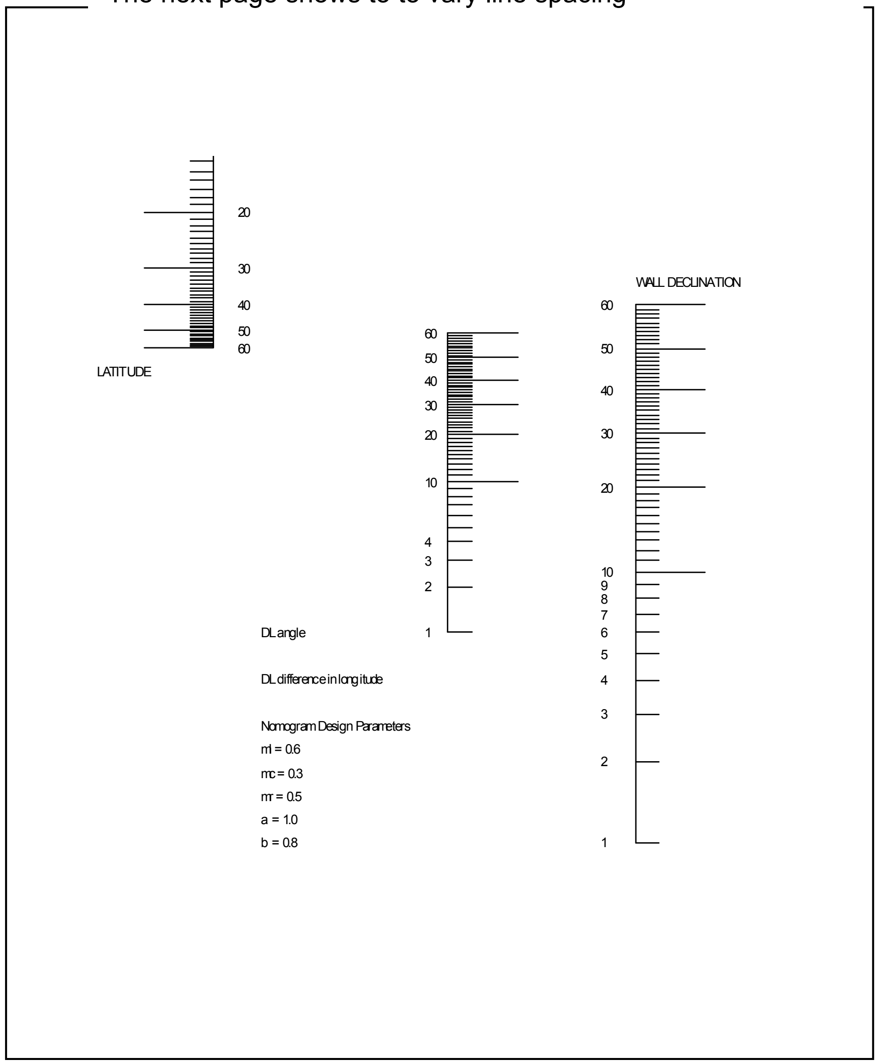
THE NEXT STEP

ALTERING NOMOGRAMS TO LOOK BETTER BY VARYING LINE SEPARATION

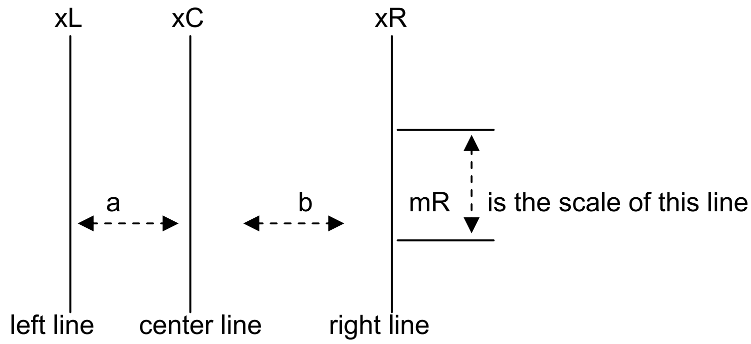
Which is the easier to read, the skinny nomogram on the left, or the better proportioned nomogram on the right.

The next page shows to to vary line spacing





The next step in advanced nomogram work is to change the line separation, and the scale of one of the lines. In other words what do you do if a nomogram looks just plain weird to make it more usable.



The “x” values are positions along the X scale, for example:-

xl = -1		
xc = 0	so	a = Abs(xl)-Abs(xc)
xr = 0.8	so	b = Abs(xr)-Abs(xc)

and the “mR” is a scale (modulus) of the right line, and between the “a” and “b” separations, and the modulus of the scale, “mR” for example we can come up with a scale (modulus) for the other two lines. For example, assume :-

mr = 0.5

in this case the right scale, we derive the modulus (scale) for the left and center lines:-

ml = Abs(mr) * a / b
mc = ml*mr / (ml+mr)

The above logic allows the scales as well as line separation to be manipulated to achieve a more usable nomogram.

To the right are a set of real parameters both entered (xL, xC, and Xr, along with mR) and the derived parameters of mL and mC.

Nomogram Design Parameters (=> means derived)

ml => 0.6
mc => 0.3
mr = 0.5
a => 1.0
b => 0.8
xl = -1.0
xc = 0.0
xr = 0.8

THE NEXT STEP

MOVING FROM THREE PARALLEL LINES

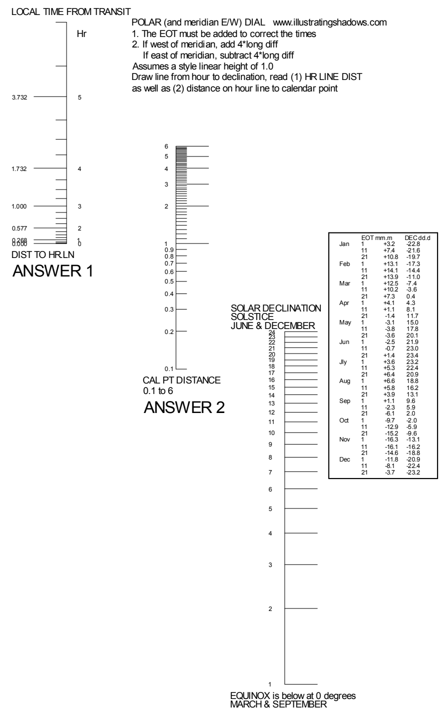
TO

AN N OR Z SHAPE

Type 2

C = L / R

Sometimes, not much can be done to better align a nomogram that uses the three parallel lines.



To the right is a normal three line nomogram for a meridian or a polar dial. For any hour from local noon (polar) or local 6 am or 6 pm (meridian) it gives the distance from the sub style to the hour line, this is a simple tan of the hour (times 15).

However, for a declination, it also gives in the center line, the distance up the hour line for that declination point. This assumes a style linear height of 1.

Fully usable, but it takes up paper space and has longer solution lines that are desirable.

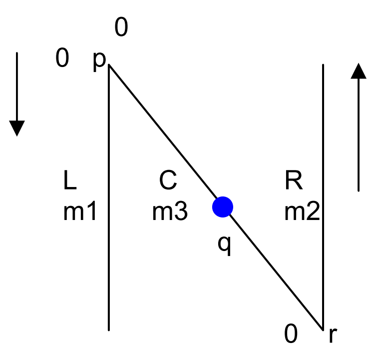
The three vertical lines works for formula:-

C = L + R and this works also for
C = L * R if logs are used

There is a nomogram chart for 1/C = 1/L + 1 / R
There is a nomogram chart for C = L / R
There is a nomogram chart for a / b = c / d

What may help here is the C = L / R nomogram, and that is called the N or Z nomogram.

[the N or Z chart] **C = L/R kind of nomogram**



pr = is total length of diagonal
pq = length along the C scale

for equations like C = L / R
L and R scales are linear, the C scale is not

pq [along the diagonal] = pr / ((m2/m1) + v) where “v” is the value of the data point

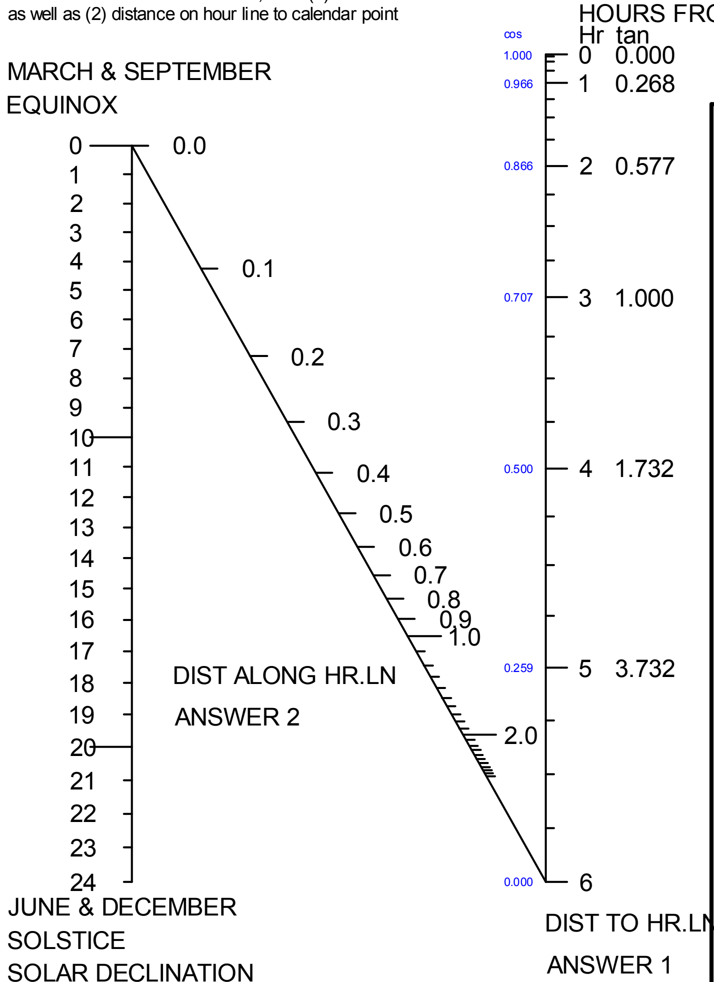
NOTE: the m2 and m1 are the scale multipliers, not the range of numbers, nor the tan or cos of a value. So for *dist=tan(dec)/cos(hr)* with dec=0 to 24, hr=0 to 6, neither the 24 nor 6 are used, nor the tan(24) nor cos(6 hours), but rather the scale multipliers if any, in this DeltaCAD macro the left scale was multiplied by 4, the right scale multiplied by 2, so m2/m1 would be 2/4 in this case.

NOTE: the left side scales have 0 at their vertex, and the diagonal connects the 0 of both vertical scales. The 0 is for the value used in the formula, so if R were cos(value) that 0 would be cos(value) (i.e. represent 90 degrees) and not “value”, see next page.

This chart is the N or Z nomogram, where the center sloped line “C” represents the left side “C” divided by the right side “R”. It does division directly, whereas the last page used logarithms to make the division a subtract and thus use the three vertical line nomogram.

Same end result but better paper usage.

POLAR (and meridian E/W) DIAL www.illustratingshadows.com
1. The EOT must be added to correct the times
2. If west of meridian, add 4*long diff
If east of meridian, subtract 4*long diff
Assumes a style linear height of 1.0
Draw line from hour to declination, read (1) HR LINE DIST
as well as (2) distance on hour line to calendar point



		EOT mm.m	DEC dd.d
Jan	1	+3.2	-22.8
	11	+7.4	-21.6
	21	+10.8	-19.7
Feb	1	+13.1	-17.3
	11	+14.1	-14.4
	21	+13.9	-11.0
Mar	1	+12.5	-7.4
	11	+10.2	-3.6
	21	+7.3	0.4
Apr	1	+4.1	4.3
	11	+1.1	8.1
	21	-1.4	11.7
May	1	-3.1	15.0
	11	-3.8	17.8
	21	-3.6	20.1
Jun	1	-2.5	21.9
	11	-0.7	23.0
	21	+1.4	23.4
Jly	1	+3.6	23.2
	11	+5.3	22.4
	21	+6.4	20.9
Aug	1	+6.6	18.8
	11	+5.8	16.2
	21	+3.9	13.1
Sep	1	+1.1	9.6
	11	-2.3	5.9
	21	-6.1	2.0
Oct	1	-9.7	-2.0
	11	-12.9	-5.9
	21	-15.2	-9.6
Nov	1	-16.3	-13.1
	11	-16.1	-16.2
	21	-14.6	-18.8
Dec	1	-11.8	-20.9
	11	-8.1	-22.4
	21	-3.7	-23.2

CIRCLES

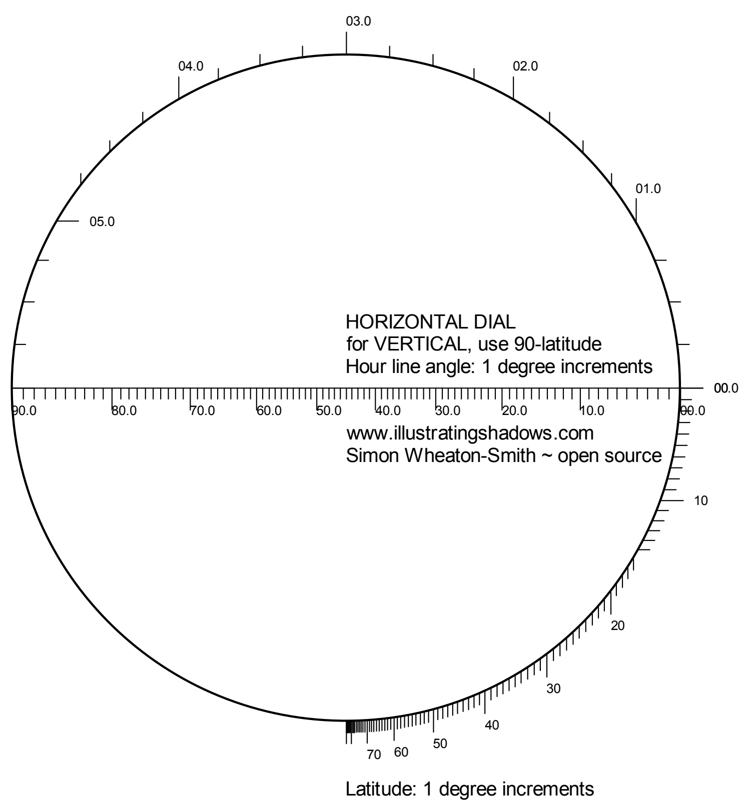
C = L * R

type 3

directly

see pages 209 and 210 of THE NOMOGRAM by Allcock and Jones

this works for well with values that on the three parallel line nomogram would reach to infinity
this also works well for formulae that are C = L * R   and are performed without needing logs



CIRCULAR NOMOGRAM FORMULA (using an h-dial as an example)

assuming: tan(hla) = sin(lat) * tan(ha) then

from xL the x value for the top HOUR ANGLE (HA is 0 to 90, and 1 hour is 15 degrees) circle is:-

x = S / (1 + tan**2(ha))

from xL the x value for the lower LATITUDE circle is:-

x = S / (1 + sin**2(lat))

from xL the x value for the horizontal HOUR LINE ANGLE line is:-

x = S / (1 + tan(hla))

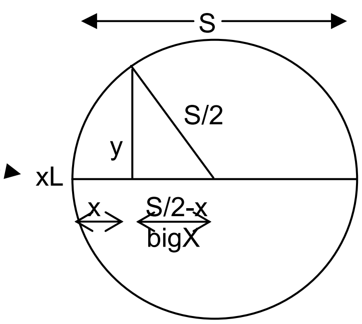
MAKING A USABLE X AND Y FOR THE CIRCLE

Having an "x" value along the horizontal line is nice, but how do we get a “y” value where a vertical line extended at "x" meets the circle

by definition, the scale "S" is the semi-circle's diameter thus

bigX = S/2 – x where X is the X from the semicircle center and
where x is the value derived above

xL is the staring reference point for “x”
for all variables



from the semi circle's center we have the x value "bigX", and we have the radius which is "S/2" thus by Pythagorus Theorum

```
(S/2)*(S/2) = bigX*bigX + y*y                 so
y*y = (S/2)*(S/2) - X*X                       but bigX = S/2 – x, so
y = sqrt ( S*S/4  - X*X )
y = sqrt ( S*S/4  - (S/2 - x)*(S/2 - x) )      multiply the two parens
y = sqrt ( S*S/4  - (S*S/4 - S*x/2 - x*S/2 + x*x) )
y = sqrt ( S*S/4  - (S*S/4 - 2*S*x/2          + x*x) )
y = sqrt ( S*S/4  - (S*S/4 - S*x          + x*x) )
y = sqrt (                 S*x          - x*x) )
y = sqrt ( S*x - x*x )
```

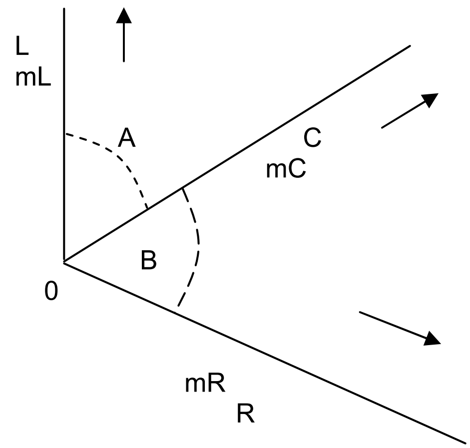
THE NEXT STEP

OTHER FORMS FOR

$1 / C = 1 / L + 1 / R$

Type 4

[the concurrent scale chart]   1/C = 1/L + 1/R kind of nomogram



for equations: 1 / L + 1 / R = 1 / C,   linear examples would be parallel resistances

the zero of all scales meets at the junction of all lines

angle A = angle B and you select the angle

you select the scale of the center line m3, then you derive m1 and m2

$m_L = m_R = m_C / (2 * \cos A)$

or you select the scale of either the left or right line m3, then you derive the others

$m_L = m_R$

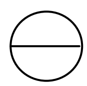
$m_C = m_L * (2 * \cos A)$

if A = B = 60 then as cos(60) = 0.5 then 2*cos(60) = 1, and all three scales are equal

EXAMPLE PROGRAM CODE FOR

Type 1: | | |

Type 2: N or Z

Type 3: 

Type 4: \/__

Both DeltaCAD and Lazarus support all the nomogram types.

Type 1 ~ THREE PARALLEL EQUIDISTANT LINE SKELETON CODE FOR DELTACAD

```
If chc = -1 Then
   ' **************************************************************************
   ' ***  -1 is test three vertical lines nomogram                          ***
   ' **************************************************************************

   ' *** Set X coordinates
   '  X = -1     latitude
   '  X =  0     resulting hour line angle
   '  X = +1     hour angle of the sun
   'Dim xl, xr, xc As Single
   'Dim x          As Single
   xl = 0.0
   xc = 0.5
   xr = 1.0

   ' set the text color, font, size, etc also
   dcSetTextParms dcBLACK,"Ariel","Bold",0,6, 20,0,0
   dcSetLineParms dcBLUE,dcSOLID,dcTHIN

   ' *** input #1 line is on the left
   dcSetTextParms dcBLACK,"Ariel","Bold",0,6, 20,0,0
   'Dim lat  As Single

   For lat = 0 to 8 step 1

       ' save y low and high values
       y = 0.2 * lat          ' use a scale factor of 0.2 to make sizes
reasonable
       If lat = 0 Then
          ylo = y
       Else
          yhi = y
       End If

       ' draw a marker line
       dcCreateText xl ,  y, 0, Format(lat, "00")
       dcSetLineParms dcBLACK,dcSOLID,dcTHIN
       dcCreateLine xl,   y, xl-0.1, y

   Next lat
   dcCreateLine xl,   ylo, xl, yhi
   dcSetTextParms dcBLACK,"Ariel","Bold",0,6, 20,0,0
   dcCreateText xl ,  yhi+0.1, 0, "IN #1"


   ' *** input #2 line is on the right
   dcSetTextParms dcBLACK,"Ariel","Bold",0,6, 20,0,0
   'Dim hr  As Single

   For hr = 0 to 8 step 1
       ' save y low and high values
       y = 0.2 * hr     ' use a scale factor of 0.2 to make sizes reasonable
       If hr = 0  Then
          ylo = y
       Else
          yhi = y
       End If

       ' draw a marker line
       ' but only say number if 10 multiple
       dcCreateText xr ,  y, 0, Format(hr, "00")
       dcCreateLine xr,   y, xr-0.1, y
```

```
   Next hr
   dcCreateLine xr,   ylo, xr, yhi
   dcSetTextParms dcBLACK,"Ariel","Bold",0,6, 20,0,0
   dcCreateText xr ,  yhi+0.1, 0, "IN #2"

   ' *** sum or solution is in the center
   dcSetTextParms dcBLACK,"Ariel","Bold",0,6, 20,0,0
   'Dim hla  As Single

   For hla = 4 to 16 step 1
       ' save y low and high values
       ' but mid scale is 0.5 of actual
       y = 0.5 * 0.2 * hla     ' the 0.5 is also on top of use a scale factor
of 0.2 to make sizes reasonable
       If hla = 4  Then
          ylo = y
       Else
          yhi = y
       End If

       ' draw a marker line
       dcCreateText xc-0.15 ,  y, 0, Format(hla, "00")
       dcSetLineParms dcBLACK,dcSOLID,dcTHIN
       dcCreateLine xc,   y, xc+0.1, y


   Next hr
   dcCreateLine xc,   ylo, xc, yhi
   dcSetTextParms dcBLACK,"Ariel","Bold",0,6, 20,0,0
   dcCreateText xc ,  yhi+0.1, 0, "IN#1 + IN#2"
   dcSetTextParms dcBLACK,"Ariel","Bold",0,6, 20,0,0
   dcCreateText xc ,  yhi+0.3, 0, "Starts at 4 only to show how all"
   dcCreateText xc ,  yhi+0.2, 0, "lines always have same base of 0"
End If
```

Type 2 ~ N OR Z NOMOGRAM SKELETON CODE FOR DELTACAD

```
If chc = 0 Then
   ' ***************************************************************************
   ' ***   0 is test N or Z nomogram                                         ***
   ' ***************************************************************************

   '   distance up an hour line to calendar pt = tan (declination) / cos(time )

   ' *** Set X coordinates
   '  X = -1
   '  X =  0
   '  X = +1
   xl = 0.00
   xc = 0.5
   xr = 1.0

   ' *** set size multiplier
   'Dim s1, s2 As Single
   s1 = 0.1
   s2 = 0.3

   ' *** set modulus
   'Dim m1, m2 As Single
   m1 = 20
   m2 = 10

   ' set the text color, font, size, etc also
   dcSetTextParms dcBLACK,"Ariel","Bold",0,4, 20,0,0
   dcSetLineParms dcBLACK,dcSOLID,dcTHIN



 ' *** SOLAR DECLINATION of the dial plate's hour lines on the left
   dcSetTextParms dcBLACK,"Ariel","Bold",0,4, 20,0,0

   ' winter
   For decl = 0 to m1 step 1
       ' save y low and high values
       ' make scale *** m2 = 2.5 ***
       'y =  - s1 * Tan(decl*2*3.1416/360)
       y =  - s1 * decl
       If decl = 0  Then
          ylo = y
       Else
          yhi = y
       End If

       ' draw a marker line
       ' but only say number if 10 multiple
       If (decl/10 - Int(decl/10)) = 0 Then
          dcCreateText xl-0.15 ,  y, 0, Format( decl, "#0")
          dcSetLineParms dcBLACK,dcSOLID,dcTHIN
          dcCreateLine xl,   y, xl-0.1, y
       Else
          dcCreateText xl-0.15 ,  y, 0, Format( decl, "#0")
          dcCreateLine xl,   y, xl-0.02, y
       End If
   Next hr
   ' *** SOLAR DECLINATION of the dial plate's hour lines on the right -
continued
   dcSetTextParms dcBLACK,"Ariel","Bold",0,4, 20,0,0
```

```
   dcCreateText xl-0.3 , yhi-0.06 , 0, "20 range"
   dcCreateText xl-0.3 , ylo+0.18 , 0, "0 range"
   ' *** SOLAR DECLINATION of the dial plate's hour lines on the right -
continued
   dcSetTextParms dcBLACK,"Ariel","Bold",0,4, 20,0,0

   dcCreateLine xl,   ylo, xl, yhi
   dcSetTextParms dcBLACK,"Ariel","Bold",0,4, 20,0,0

   ' save the line bottom for the N or Z line
   'Dim xxxx,yyyy As Single
   'Dim yyyyy As Single
   xxxx = xl
   yyyy = ylo
   yyyyy=yhi



   ' *** a value line is on the right of "X" which becomes 1/X because
   ' the nomogram implies the 1/X
   dcSetTextParms dcBLACK,"Ariel","Bold",0,4, 20,0,0
   For hr = 0 to m2 step  0.25
       ' save y low and high values
       y = yyyyy + s2 *  hr
       If hr = 0 Then
          ylo = y
       Else
          yhi = y
       End If

       ' draw a marker line
       ' but only say number if 10 multiple
       If (hr  - Int(hr)) = 0 Then
          dcCreateText xr+0.1 ,    y, 0, Format(hr, "#0")
          dcSetLineParms dcBLACK,dcSOLID,dcTHIN
          dcCreateLine xr,     y, xr+.1, y
       Else
          dcCreateLine xr,     y, xr+.02, y
       End If
   Next hr
   dcCreateLine xr,   ylo, xr, yhi
   dcSetTextParms dcBLACK,"Ariel","Bold",0,4, 20,0,0
   dcSetTextParms dcBLACK,"Ariel","Bold",0,4, 20,0,0
   dcCreateText xr+0.1 ,  yhi-0.1, 0, "0 to 10"

   ' save the line bottom for the N or Z line
   'Dim xxx,yyy As Single
   xxx = xr
   yyy = ylo



   ' *** DISTANCE ON CENTER SCALE IS LEFT/RIGHT
   '
   'Dim pr As Single
   'Dim pq As Single
   ' pr is the length of the diagonal
   pr = Sqr( (xxxx-xxx)*(xxxx-xxx) + (yyyy-yyy)*(yyyy-yyy) )


   dcSetTextParms dcBLUE,"Ariel","Bold",0,4, 20,0,0
   dcSetLineParms dcBLUE,dcSOLID,dcTHIN
   dcCreateLine xxx,yyy,xxxx,yyyy
```

```
   'Dim decdist As Single
   For decpt = 0 to 20  step 1
       'decpt = decdist/10
       ' get point along the diaganol ~ Z = L f3(w) / [(m2/m1) + f3(w)]
       pq = ( pr * decpt ) / ((s2/s1) + decpt)

       ' need an x and a y for that distance
       x = xxxx + ((xxx-xxxx) * (pq/pr))
       y = yyyy + ((yyy-yyyy) * (pq/pr))


       ' draw a marker line but only say number if integer
       If (decpt/10 - Int(decpt/10)) = 0 Then
          dcCreateText x+0.1  ,  y, 0, Format(decpt , "#0")
          dcCreateLine x,   y, x+0.2, y
       ElseIf decpt <10 Then
          dcCreateLine x,   y, x+0.1, y
          dcCreateText x+0.1  ,  y, 0, Format(decpt , "0.0")
       Else
          dcCreateLine x,   y, x+0.1, y
       End If
   Next decpt
   dcSetTextParms dcBLACK,"Ariel","Bold",0,4, 20,0,0
   dcSetLineParms dcBLACK,dcSOLID,dcTHIN
   dcCreateText xc-0.4  ,  ylo+0.8, 0, "LEFT/RIGHT"

End If
```

Type 3 ~ CIRCULAR SKELETON CODE FOR DELTACAD

```
' If chc = 1 Then
   ' ***********************************************************************
   ' *** horizontal dial nomogram itself   AS A CIRCLE NOMOGRAM             ***
   ' ***********************************************************************


   ' *********************************
   ' * circular nomogram key points *
   ' *********************************

   Dim S As Single
   ' S  ~ ~ ~ ~ ~ ~ the scale multiplier
   S = 3

   ' *** Set X coordinates
   Dim xl, xr, xc As Single
   Dim x, bigX    As Single
   xl = -1
   'xc = xL + S/2
   'xr = xC + S/2

   ' *** Set Y coordinates
   Dim yDisp As Single
   yDisp = 0

   ' *********************************
   ' * circular nomograms           *
   ' *********************************

   ' xL ~ ~ ~ ~ ~ ~ the focal point of all scales for a circular nomogram

   ' *********************************
   ' * circular nomogram formula    *
   ' *********************************

   ' assuming:  C = L * R   then
   '
   ' from xL the x value for the bottom semicircle is:-
   '
   ' x = S / ( 1 + L * L )
   '
   ' from xL the x value for the top semicircle is:-
   '
   ' x = S / ( 1 + R * R )
   '
   ' from xL the x value for the answer line is:-
   '
   ' x = S / ( 1 + C )
   '
   ' see pages 209 and 210 of THE NOMOGRAM by Allcock and Jones

   ' *******************************************
   ' * making a usable X and Y for the circle *
   ' *******************************************

   ' having an "x" value along the horizontal line is nice, but how do
   ' we get a Y value where a vertical line extended at "x" meets the circle
   '
   ' by definition, the scale "S" is the semi-circle's diameter thus
   '
   ' X = S/2 - x        where X is the X from the semicircle center and
   '                    where x is the value derived above
   '
   '
   '              +
   '         +          +
   '       +    S/2       +
   '      +  Y             +
   '      *..+....C........*                    length "*" to "*" is S
   '       <x>                                  S/2 is radius and hypotenuse
```

```
'        <  X >
'
' from the semi circle's center "C" we have the x value "X"
' and we have the radius which is "S/2"
'
' thus by Pythagorus Theorum
'
' (S/2)*(S/2) = X*X + Y*Y
'
' or
'
' Y*Y = (S/2)*(S/2) - X*X
'
' but X = S/2 - x
'
' Y = sqrt ( S*S/4  - X*X )
'
'   = sqrt ( S*S/4  - (S/2 - x)*(S/2 - x) )    now multiply the two parenthesis
'
'   = sqrt ( S*S/4  - (S*S/4 - S*x/2 - x*S/2 + x*x) )
'
'   = sqrt ( S*S/4  - (S*S/4 - 2*S*x/2       + x*x) )
'
'   = sqrt ( S*S/4  - (S*S/4 - S*x           + x*x) )
'
'   = sqrt ( S*S/4  -  S*S/4 + S*x           - x*x) )
'
'   = sqrt (                   S*x           - x*x) )
'
'   = sqrt ( S*x - x*x )


' draw the two semi circles and the line
dcCreateCircle    xL+S/2 , y  , S/2
dcCreateLine      xL     , 0  , xl+S, 0

' set the text color, font, size, etc also
dcSetTextParms dcBLACK,"Ariel","Bold",0,4, 20,0,0
dcSetLineParms dcBLUE,dcSOLID,dcTHIN



' *** L is on the bottom
'
' x = S / ( 1 + L * L )
'
' and
'           X*X means the square of the hypotenuse
' Y = sqrt ( S*x - x*x )
'             x here is S/2 - x
'
dcSetTextParms dcBLACK,"Ariel","Bold",0,4, 20,0,0
Dim L  As Single

For L = 0 To 10 Step 1

    ' get x from xL distance
    x    = S / ( 1 + (L*L) )

    ' bigX is x from circle center and is used for Y calculation
    bigX = S/2 - Abs(x)               ' x may have been L or R of center
    Y = Sqr ( s*x - x*x )

    ' get x and y for screen location ~ i.e. adjust by yDisp and xL
    x = xL    + x
    y = yDisp + y

    ' draw a marker line and number
    If Abs(y) < S/4 Then
          dcCreateText x+0.15  ,  -y , 0, Format(L, "00")
          dcSetLineParms dcBLACK,dcSOLID,dcTHIN
          dcCreateLine x ,   -y, x+0.1 , -y
```

```
    End If
    If Abs(y) >= S/4 Then
          dcCreateText x  ,  -y-0.15 , 0, Format(L, "00")
          dcSetLineParms dcBLACK,dcSOLID,dcTHIN
          dcCreateLine x ,   -y, x , -y-0.1
    End If
Next lat
dcSetTextParms dcBLACK,"Ariel","Bold",0,6, 20,0,0
dcCreateText      xL + S/2 ,  -S/2-0.3, 0, "L: bottom: 1 to 10"


' *** R is on the top
'
' x = S / ( 1 + R*R )
'
' and
'           X*X means the square of the hypotenuse
' Y = sqrt ( S*x - x*x )
'             x here is S/2 - x
'
dcSetTextParms dcBLACK,"Ariel","Bold",0,4, 20,0,0
Dim R  As Single

For R = 0 To 5 Step 1

    ' get x from xL distance (ha here is 4 times hour thus ha*0.25)
    '                        (and of course 1 hour is 15 degrees)
    x    = S / ( 1 + (R*R))

    ' bigX is x from circle center and is used for Y calculation
    bigX = S/2 - Abs(x)               ' x may have been L or R of center
    Y = Sqr ( s*x - x*x )

    ' get x and y for screen location ~ i.e. adjust by yDisp and xL
    x = xL    + x
    y = yDisp + y

    ' draw a marker line but only say number if 10 multiple
    If Abs(y) < S/4 Then
          dcCreateText x+0.15  ,  y , 0, Format(R, "00")
          dcSetLineParms dcBLACK,dcSOLID,dcTHIN
          dcCreateLine x ,   y, x+0.1 , y
    End If
    If Abs(y) >= S/4 Then
          dcCreateText x  ,  y+0.15, 0, Format(R, "00")
          dcSetLineParms dcBLACK,dcSOLID,dcTHIN
          dcCreateLine x ,   y, x , y+0.1
    End If
Next lat
dcSetTextParms dcBLACK,"Ariel","Bold",0,6, 20,0,0
dcCreateText      xL + S/2 ,   S/2+0.3, 0, "R: top: 1 to 5"


' *** C is the answer in the center line
'
' x = S / ( 1 + C )
'
'              not tan*tan for the horizontal line
'
dcSetTextParms dcBLACK,"Ariel","Bold",0,4, 20,0,0
Dim C  As Single

For C = 0 To 50 Step 1

    ' get x from xL distance
    x    = S / ( 1 +  C  )

    Y = 0

    ' get x and y for screen location ~ i.e. adjust by yDisp and xL
    x = xL    + x
    y = yDisp + y
```

```
        ' draw a marker line
        ' but only say number if 10 multiple
        If (hla/10 - Int(hla/10)) = 0 Then
           dcCreateText x  ,   y-0.1, 0, Format(C, "00")
           dcSetLineParms dcBLACK,dcSOLID,dcTHIN
           dcCreateLine x ,    y    , x , -y-0.1
        Else
           dcCreateLine x ,    y    , x , -y-0.05
        End If

    Next hla
    dcSetTextParms dcBLACK,"Ariel","Bold",0,6, 20,0,0
    dcCreateText      xL + S/2 ,   0+0.1, 0, "C=L*R: center: 1 to 50"

    dcCreateText      xL + S/2 ,   0+0.3, 0, "CIRCULAR NOMOGRAM"
    dcCreateText      xL + S/2 ,   0+0.2, 0, "for C = L * R without logs"
    dcCreateText      xL + S/2 ,   0-0.2, 0, "www.illustratingshadows.com"
    dcCreateText      xL + S/2 ,   0-0.3, 0, "Simon Wheaton-Smith ~ open source"

'End If
```

Type 4 ~ ANGULAR TYPE 4 NOMOGRAM SKELETON CODE IN LAZARUS

```
{ ******************************************************************** }
{ ***    S T A R T    O F   T E S T   T Y P E   3              *** }
{ ******************************************************************** }

procedure TForm1.testType3Click(Sender: TObject);
{
     KEY POINTS TO REMEMBER:
     0,0   top left
     y increases positively down

     DOCUMENTATION:
     http://delphi.about.com/library/bluc/text/uc052102a.htm     [ is page 1 ]
     http://delphi.about.com/library/bluc/text/uc052102b.htm     [ is page 2 ]
}

var { must declare FOR variables in the section in which they are used }

    { x and y working values }
    x,y,aspect,angle :single  ;

begin

  { ***
    *** SET NORMAL PARAMETERS
    ***
  }
  { left, center, and right X placement for the lines }
  xl     :=    100+strtoint(bValue.text) ;

  { angle between the three lines }
  angle  :=    strtoint(vAngle.text) ;



  { the xybase (baseline for Y) is xysize }
  xybase :=   xysize-100;

  {  *** set size multiplier ~ and this is used in the center scale answer }
  sL :=15.0  * strtoint(mRvalue.text) / 10;     // left      <<<<<  varies
  sR :=sL ;                                     // right     <<<<<  fixed
  // sL = sR = sC / (2 cos Angle)    thus       sC = sL * (2 cos Angle)
  sC := sL * (2 * cos(degtorad*angle));

  { advise about good values }
  recmRvalue.caption   := '10';
  recDisplaceY.caption := '-150';
  recBvalue.caption    := '150';

  { ***
    *** INITIAL CANVAS SCREEN SETUP
    ***
  }
  begin
     { see FormPaint }
     { http://delphi.about.com/library/bluc/text/uc052102d.htm }

     { clear the graph area see: http://delphi.about.com/library/bluc/text/uc052102c.htm}
     Canvas.Rectangle( Bounds(xshift-15, yshift-15, 30+xySize, 30+xySize));

     { say what is being depicted }
     Canvas.TextOut  ( xshift,      yshift-14   ,'TYPE 4 TEST');

     { move to aValue starting point if 0,0 (top left) scales shifted of course }
     Canvas.MoveTo   ( xshift ,      yshift      );

     { draw the inner box box which lines will be bounded by }
     Canvas.Pen.Color :=  100;   { dark brown }
     Canvas.Pen.Width :=    1;
     Canvas.LineTo    ( xshift+xySize,    yshift      );
     Canvas.LineTo    ( xshift+xySize,    yshift+xySize );
```

```
    Canvas.LineTo    ( xshift ,          yshift+xySize );
    Canvas.LineTo    ( xshift ,          yshift      );

    { move pen to top and center }
    Canvas.Pen.Width :=    1;
    Canvas.Pen.Color :=  255;
 end ;

 { ***                                      \
   *** F I R S T ***   DRAW THE LEFT LINE    \
   ***                                        \
 }
 begin
    for hrI  := 0 to 20 do
    begin
       hr  := hrI ;
       { get an X }
       // cos 60 = 0.5  and sin 60 = 0.866
       x   := xL +sL * hr * cos(degtorad*angle*2) ;

       { get a Y }
       y   := -sL * hr * sin (degtorad*angle*2)+ strtoint(yDisp.text);

       { draw a horizontal marker for this latitude }
       Canvas.MoveTo   ( xshift+   getGrXY(x),  yshift+xybase+getGrXY(y)      );
       Canvas.LineTo   ( xshift-10+getGrXY(x),  yshift+xybase+getGrXY(y)      );

       { label the horizontal marker line }
       if (hr < 11) or (hr = 15) or (hr = 20) then
       begin
          Canvas.TextOut      (    xshift-20+getGrXY(x),      yshift+xybase-8+getGrXY(y),
FloatToStrF (hr,ffFixed,2,0) );
       end ;
    end ;

    { draw the actual nomogram line }
    x   := xL +sL * 0 * cos(degtorad*2*angle) ;
    y   :=    -sL * 0 * sin (degtorad*2*angle)+ strtoint(yDisp.text);
    Canvas.MoveTo   ( xshift+   getGrXY(x),yshift+xybase+getGrXY(y) );
    //              XX <---- the 0 and the 20 are the scale limits
    x   := xL +sL * 20 * cos(degtorad*2*angle) ;
    y   :=    -sL * 20 * sin (degtorad*2*angle)+ strtoint(yDisp.text);
    Canvas.LineTo   ( xshift+   getGrXY(x),yshift+xybase+getGrXY(y) );

    { give line a heading }
    Canvas.TextOut  ( xshift+   getGrXY(x),yshift+xybase+getGrXY(y)-30,   'LEFT 0:20');

 end ;

 { ***                                      /
   *** SECOND ***  DRAW THE CENTER LINE    /
   ***                                    /
 }
 begin
    for hrI  := 0 to 20 do
    begin
       hr  := hrI ;
       { get an X }
       x   := xL +sC * hri * cos(degtorad*angle) ;             { x is L line X }

       { get a Y }
       y   := -sC * hr * sin (degtorad*angle)+ strtoint(yDisp.text);       { multiply by
modulus or scale }

       { draw a horizontal marker for this latitude }
       Canvas.MoveTo   ( xshift+   getGrXY(x),  yshift+xybase+getGrXY(y)      );
       Canvas.LineTo   ( xshift-10+getGrXY(x),  yshift+xybase+getGrXY(y)      );

       { label the horizontal marker line }
       Canvas.TextOut   (  xshift-20+getGrXY(x),   yshift+xybase-8+getGrXY(y), FloatToStrF
(hr,ffFixed,2,0) );
    end ;
```

```
     { draw the actual nomogram line }
     x   := xL +sC * 0 * cos(degtorad*angle) ;
     y   :=    -sC * 0 * sin (degtorad*angle)+ strtoint(yDisp.text);
     Canvas.MoveTo   ( xshift+   getGrXY(x),yshift+xybase+getGrXY(y) );
     //              XX <---- the 0 and the 20 are the scale limits
     x   := xL +sC * 20 * cos(degtorad*angle) ;
     y   :=    -sC * 20 * sin (degtorad*angle)+ strtoint(yDisp.text);
     Canvas.LineTo   ( xshift+   getGrXY(x),yshift+xybase+getGrXY(y) );

     { give line a heading }
     Canvas.TextOut    (  xshift+      getGrXY(x),yshift+xybase+getGrXY(y)-30,      'CENTER
0:20');

  end ;


  { ***
    *** THIRD ***  DRAW THE RIGHT LINE   \  /
    ***                                      ____________

    m1 = m2 = m3 / (2 cos A)

  }
  begin
     for hrI  := 0 to 20 do
     begin
        hr  := hrI ;
        { get an X }
        x   := xL +sL * hri * cos(degtorad*0) ;             { x is L line X }

        { get a Y }
        y    := -sL  * hr  * sin  (degtorad*0)+  strtoint(yDisp.text);         {  multiply  by
modulus or scale }

        { draw a horizontal marker for this latitude }
        Canvas.MoveTo   ( xshift+   getGrXY(x),  yshift-5+xybase+getGrXY(y)      );
        Canvas.LineTo   ( xshift+   getGrXY(x),  yshift+5+xybase+getGrXY(y)      );

        { label the horizontal marker line }
        if (hr < 11) or (hr = 15) or (hr = 20) then
        begin
           Canvas.TextOut    (  xshift+      getGrXY(x),    yshift+10+xybase-8+getGrXY(y),
FloatToStrF (hr,ffFixed,2,0) );
        end ;
     end ;

     { draw the actual nomogram line }
     x   := xL +sL * 0 * cos(degtorad*0) ;
     y   :=    -sL * 0 * sin (degtorad*0)+ strtoint(yDisp.text);
     Canvas.MoveTo   ( xshift+   getGrXY(x),yshift+xybase+getGrXY(y) );
     //              XX <---- the 0 and the 20 are the scale limits
     x   := xL +sL * 20 * cos(degtorad*0) ;
     y   :=    -sL * 20 * sin (degtorad*0)+ strtoint(yDisp.text);
     Canvas.LineTo   ( xshift+   getGrXY(x),yshift+xybase+getGrXY(y) );

     { give line a heading }
     Canvas.TextOut  ( xshift+   getGrXY(x),yshift+xybase+getGrXY(y)-30,   'RIGHT 0:20');

  end ;

end;
```

COMPARING CAD SYSTEMS

There is a lot of talk about CAD systems. In the context of sundials, there are probably two main uses for CAD. The first is drafting dial plates. The second is the automation of drafting dial plates. CAD systems are available free, and they can be purchased. This section looks at a variety of these options.

DELTACAD ~ available for purchase ~ probably the best choice for programming ~ DeltaCAD is probably the most common CAD system for sundial work. It is programmed with a BASIC like language, thus there are many BASIC DeltaCAD sundial programs out there, called macros. DeltaCAD can also be used for hand drafting, which it does well, however sometimes measuring an angle will cause frustration.

www.deltacad.com

TURBOCAD ~ available for purchase ~ probably the best choice for drafting ~ TurboCAD is probably not used much in the sundial community. For programmed dial plate construction, there are two methods available. One is using an established language such as VBS. In order to use TurboCAD programs however, the Professional version must be acquired which is the most expensive, and top of the line. Extra software may be required to be installed. And the programs are slow. The other method is the parametric part scripting. Here, a sundial plate is designed with an attached script, and when the dial plate is clicked, the parameters can be changed and the dial plate will redraw itself. This innovative technique has limitations since descriptive data is very hard to add to a display, for example hour line angles. Where TurboCAD excels is in its very easy drafting abilities, and in its 3D modeling features. Almost all the figures in the Illustrating Shadows series of books were drawn using TurboCAD.

http://www.turbocad.com/

NANOCAD ~ probably the second best choice for programming ~ NanoCAD is available free. This is well designed for drafting, and the scripting or macro system uses VBS and works very well, as well as Java Script. While the documentation is cryptic, and Programming Shadows is essential to provide insights in its use, this is an excellent second choice for programming. Installs needing a free serial number which is automated along with the free license, also automated. No extra add-on software is needed.

www.nanocad.com

FREECAD ~ open source ~ probably the third best choice for programming ~ FreeCAD is part of the open source concept, and is available free. This is not very well designed for drafting, for example, drawing a line at a specific angle presents problems. However, the scripting or macro system uses Python and works very well. While the documentation is cryptic, and Programming Shadows is essential to provide insights in its use and debugging, this is an excellent second choice for programming.

http://www.freecadweb.org/

PROGECAD ~ available free for private use ~ purchased if used commercially ~ ProgeCAD downloads and installs easily, requires registration which is now very fast, and for private use is free. If it is to be used commercially, then it must be purchased. The programming method employed uses LISP as its language with the desire of providing some AutoCAD compatibility. LISP is not the most intuitive choice, however it works.

http://www.progesoft.com/en/products/progecad-smart/

POWERDRAW ~ available free ~ Powerdraw downloads and installs easily. The programming method employed uses an apparent subset of Pascal as its language. The documentation is sparse, and an article in Supplemental Shadows describes the programming of missing functionality, such as SIN, COS, TAN, ATAN, and SQRT.

www.powerdraw.software.informer.com

BLENDER ~ available free ~ Blender is a free CAD system whose focus is 3D objects. Blender supports programming by using Python scripts, however, their focus is on 3D objects and not on drafting such as is used in sundial design.

www.blender.org

PROGRAMMING FOR SUNDIAL PLATES

Illustrating Shadows has programs, scripts, or macros for DeltaCAD (probably the best choice), and for NanoCAD and FreeCAD (probably the second and third best choices).

While Illustrating Shadows does provide working macros, scripts, or programs for TurboCAD, ProgeCAD, Powerdraw, and Blender, these are probably not ranked as the best choice for the sundial designer.

DRAFTING SUNDIAL PLATES BY HAND

Which is best for hand drafting is probably a matter of personal choice. Probably TurboCAD is the cleanest drafting choice, and is certainly very well stocked with drafting tools. DeltaCAD would be a close second.

While ProgeCAD, and FreeCAD can be used for drafting, they would be a third choice, and Blender, being a 3D system, would not rank as a highly usable tool.

“WHAT ABOUT ME” CAD SYSTEMS

There are many CAD systems out there. CAD systems that can be purchased would obviously include AutoCAD, however that is a system beyond the price range of many sundial designers. CAD systems that are free tend to be harder to use for drafting. There are so many free CAD systems available, many of which the author has tried, however they usually lack programming facilities, and that lack is why the author has spent little time with them.

Scientific systems such as Octave, Scilab, and Euler are programmable, and Illustrating Shadows provides sundial programs for those systems. They would probably not be ranked in the top tier for practical sundial design.

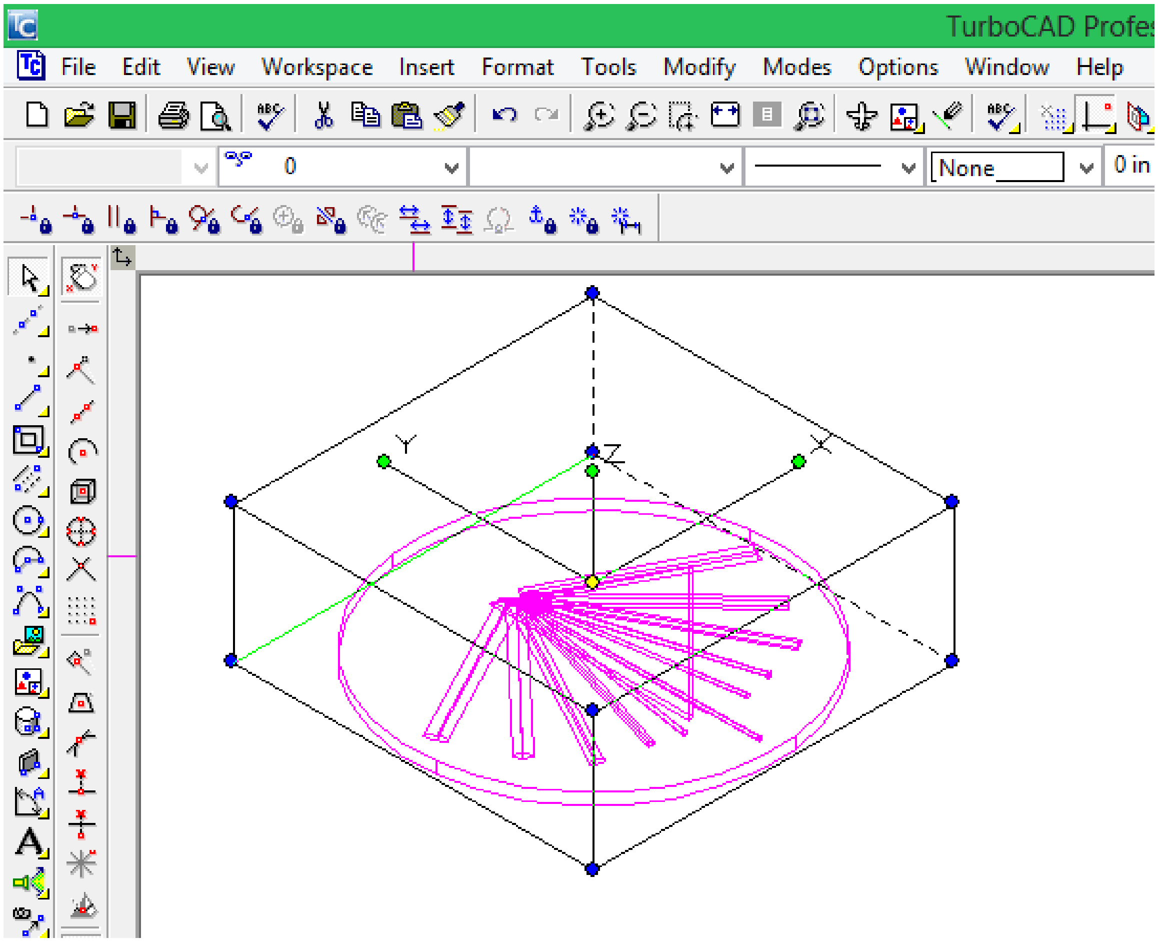
SYSTEMS SUPPORTED

The above CAD systems have been tested using Windows XP with various service packs, Windows Vista, Windows 7, and Windows 8. Also they were run in 32 and 64 bit systems. The 32 bit systems run on 64 bit systems but not the other way around.

A 3D PRINTED SUNDIAL

A simple sundial was created in CAD software, in this case TurboCAD, and it wasn't the latest version.

- 3dSundialLat33.5Lng0gnomonLinesAndPlate.tcw



This was a small sundial, so it was NOT longitude corrected. My practice is not to correct for longitude if the dial is small or portable, and to only correct for longitude when the dial is large or permanent. Of course other facts may change that decision.

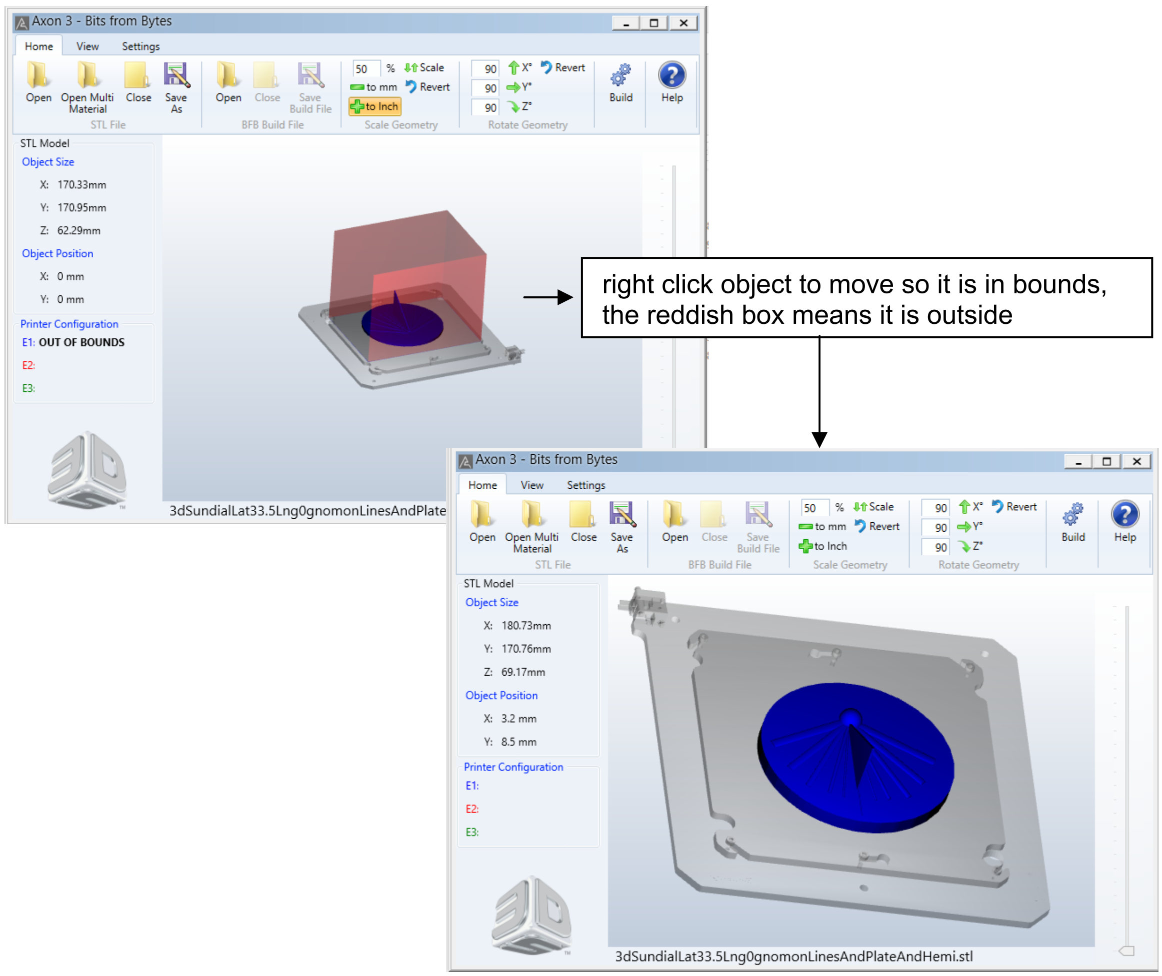
This dial was correct for latitude.

It was created in a very few simple steps as a 3d .TCW file, and then saved as a .STL file (stereo lithographic) which is a common method of feeding 3D printing systems. In essence, DeltaCAD was used to get the hour lines, that screen was inserted into TurboCAD, and then TurboCAD hour lines built over the copied DeltaCAD ones. Of course, the TurboCAD VBS scripts could have been used but that requires the Professional version. Then 3d hour lines were generated in TurboCAD, and the original 2d lines as well as the DeltaCAD depiction were removed. The gnomon was added as well as the dial plate, and an aesthetic dial center hemisphere.

The TurboCAD .TCW file was then saved as a .STL file:-

- 3dSundialLat33.5Lng0gnomonLinesAndPlate.stl

It was then viewed in an STL viewer such as Axon just to verify the conversion from TCW to STL.



This Axon viewer was downloaded from the web site:-

- http://www.bitsfrombytes.com/

The file was called:-

- Axon 3.0 Alpha 3 Setup.zip

There is a manual online at:-

- http://3dpedia.3dsystems.com/display/BFBAXON2/Axon+2+manual

This manual describes the above software and also describes how to BUILD the final file for actual printing on the final 3d printer, IF the file is to be 3d printed on your 3d printer. For using remote or cloud 3d printers, the .STL file is what is used as that is device independent.

At this point there were three choices.

1. To buy a 3d printer and install its software
2. To use a 3d printer that also does cloud printing of 3d objects, or
3. To select a local provider.

CHOICE 1:

A 3d printer can be acquired for around $1299, namely the “Cube”

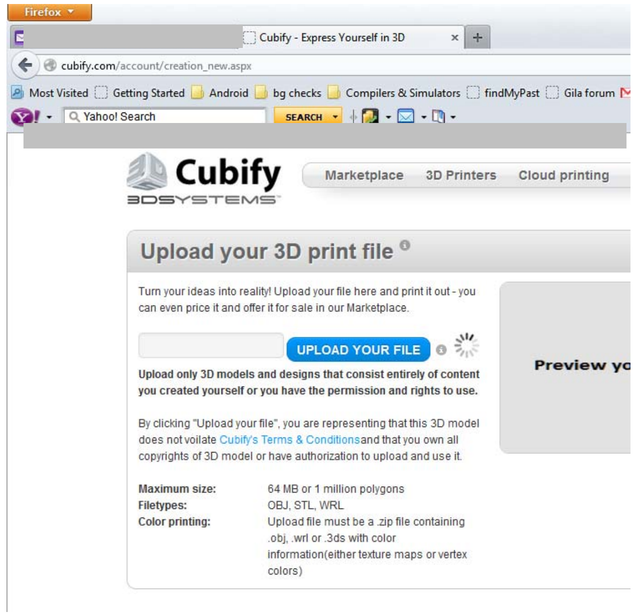
- http://cubify.com/cube/index.aspx

Print size	5.5" x 5.5" x 5.5" or 140 x 140 x 140mm
Print resolution	200 microns (0.2mm)*
Colors available	16 colors, 2 are glow in the dark
	Cube
Colors per print	✔ (1)
Prints in ABS plastic	✔
Prints in PLA plastic	✔ *

Given the price, I decided to opt for choice 2, namely to use this vendor’s cloud printing service.

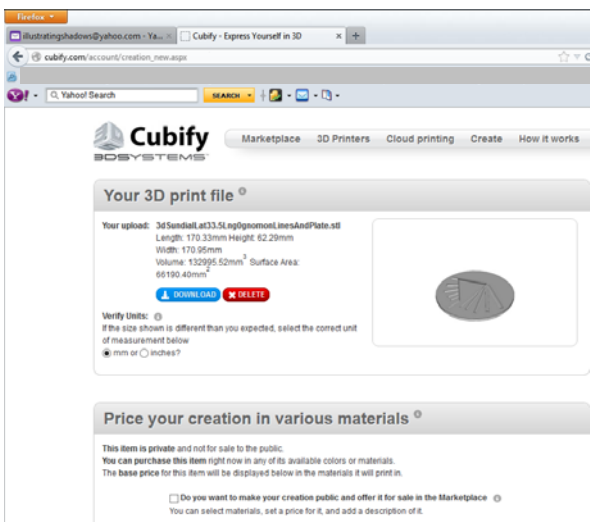
CHOICE 2:

I created an account on Cubify which was simple, and then selected cloud printing. This would help me decide whether I wanted my own 3d printer.



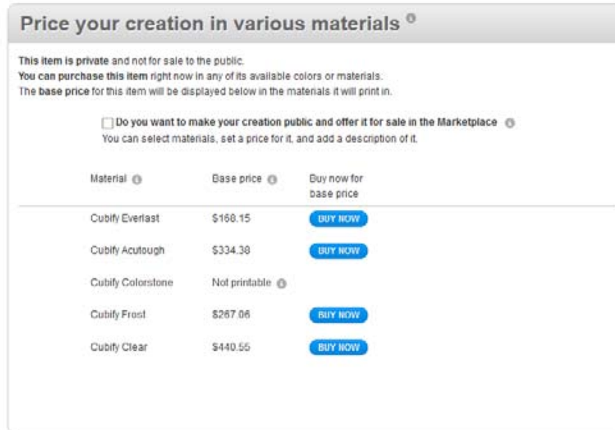
I selected a file to upload:

- 3dSundialLat33.5Lng0gnomonLinesAndPlate.stl



and it took a long time to process. Having no idea why, I went into Cubify and added a few, very few, profile details about me and tried the upload again. When I retried the upload, it worked quickly.

I had the option to print it on the cloud:-



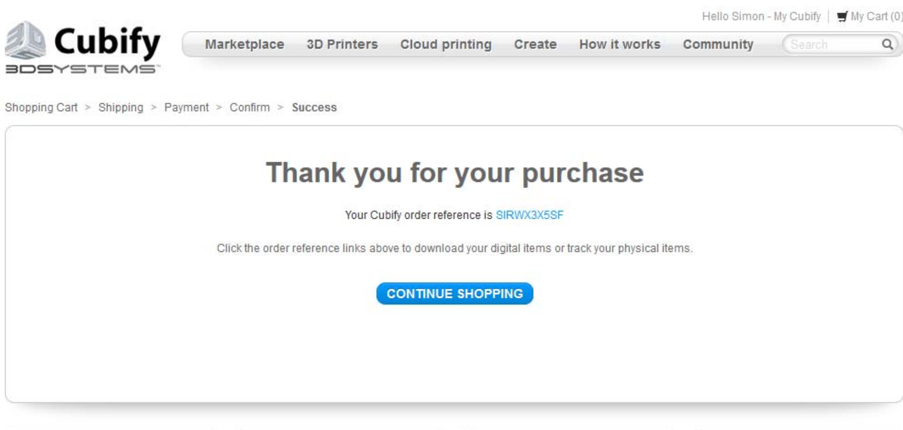
Price your creation in various materials

This item is private and not for sale to the public.
You can purchase this item right now in any of its available colors or materials.
The **base price** for this item will be displayed below in the materials it will print in.

☐ **Do you want to make your creation public and offer it for sale in the Marketplace**
You can select materials, set a price for it, and add a description of it.

Material	Base price	Buy now for base price
Cubify Everlast	$168.15	BUY NOW
Cubify Acutough	$334.38	BUY NOW
Cubify Colorstone	Not printable	
Cubify Frost	$267.06	BUY NOW
Cubify Clear	$440.55	BUY NOW

So I followed the process and made the order:-



Hello Simon - My Cubify | My Cart (0)

Cubify 3DSYSTEMS

Marketplace 3D Printers Cloud printing Create How it works Community

Shopping Cart > Shipping > Payment > Confirm > **Success**

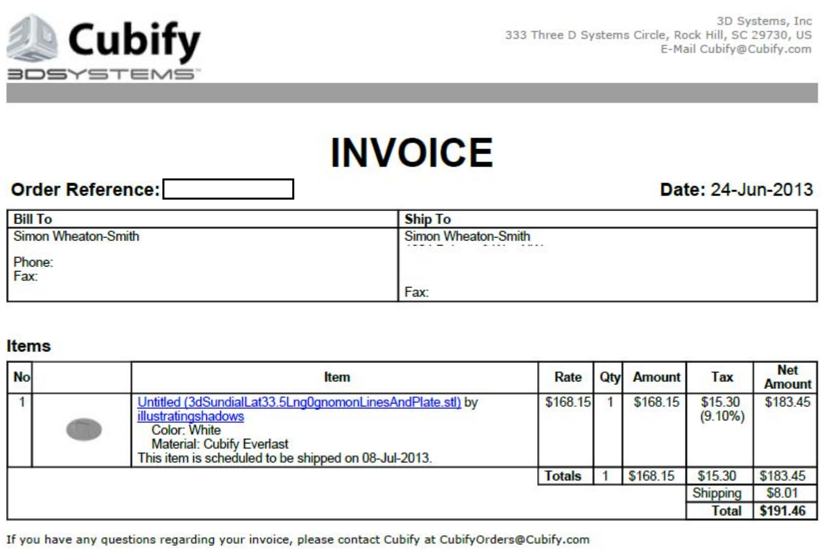
Thank you for your purchase

Your Cubify order reference is SIRWX3X5SF

Click the order reference links above to download your digital items or track your physical items.

CONTINUE SHOPPING

And then I waited for the sundial to arrive! They emailed me an invoice, and the shipping date was 15 days from the order.



Cubify 3DSYSTEMS

3D Systems, Inc
333 Three D Systems Circle, Rock Hill, SC 29730, US
E-Mail Cubify@Cubify.com

INVOICE

Order Reference: **Date:** 24-Jun-2013

Bill To	Ship To
Simon Wheaton-Smith Phone: Fax:	Simon Wheaton-Smith Fax:

Items

No		Item	Rate	Qty	Amount	Tax	Net Amount
1		Untitled (3dSundialLat33.5Lng0gnomonLinesAndPlate.stl) by illustratingshadows Color: White Material: Cubify Everlast This item is scheduled to be shipped on 08-Jul-2013.	$168.15	1	$168.15	$15.30 (9.10%)	$183.45
			Totals	1	$168.15	$15.30	$183.45
						Shipping	$8.01
						Total	$191.46

If you have any questions regarding your invoice, please contact Cubify at CubifyOrders@Cubify.com

CHOICE 3:

The other option was to use a local printer in my town. The local provider was just a few short miles away from me and was:

- http://www.padtinc.com/

Who asked for a .STL file which I sent. When converting .TCW to .STL files, sometimes scaling was a problem. However the aspect ratios were retained. A key point with TurboCAD is that ASPECT RATIO retention is NOT the default, so when rescaling anything, right click the objects, and turn on ASPECT RATIO. The printers needed to know the color, size, and purpose of the object. This affects the material to be used.

Same day I received a quote for a 4 to 5 day from receipt of order delivery date. It was actually ready the next day.

The final dial was ready the next day, so I picked it up and tested at 1031 mst. It shows, correctly, 1000 because this dial was designed for a longitude 28 minutes east and an EOT of about 3 minutes plus. So at 1031 mst the dial should shows 1000 local apparent time.

The material considerations consist of UV tolerance, and the like. The 3d print file is standardized to .STL and 3d printer vendors provide conversion software to convert it for the final print run to their internal formats. TurboCAD Deluxe for under $150 supports .STL files, as do some free programs.

A 3d printer printed dial could be intricate and used as the basis for a mold to cast a final dial.

ADDITIONAL NOTES:

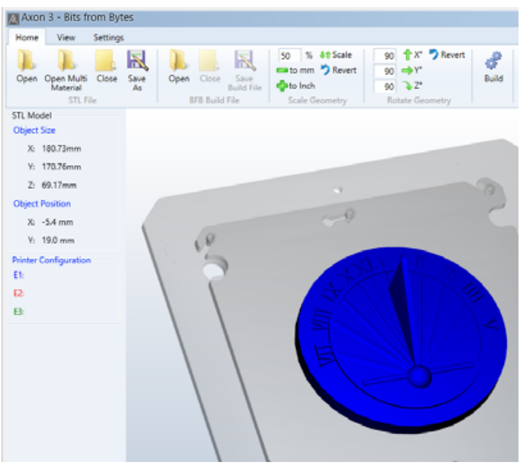
NOTES:-PLA Material ~ PLA (Polylactic Acid) may be the easiest material to work with, it is a biodegradable thermoplastic derived from renewable resources such as corn starch and sugar canes. This makes PLA environmentally friendly and very safe to work with.

ABS 3D Printing Material ~ ABS (Acrylonitrile butadiene styrene) may be the second easiest material to work, it is an engineering polymer commonly used to produce car bumpers due to its toughness and strength. It’s also the stuff that Lego blocks are made of…tough enough but safe enough for the kids to handle!

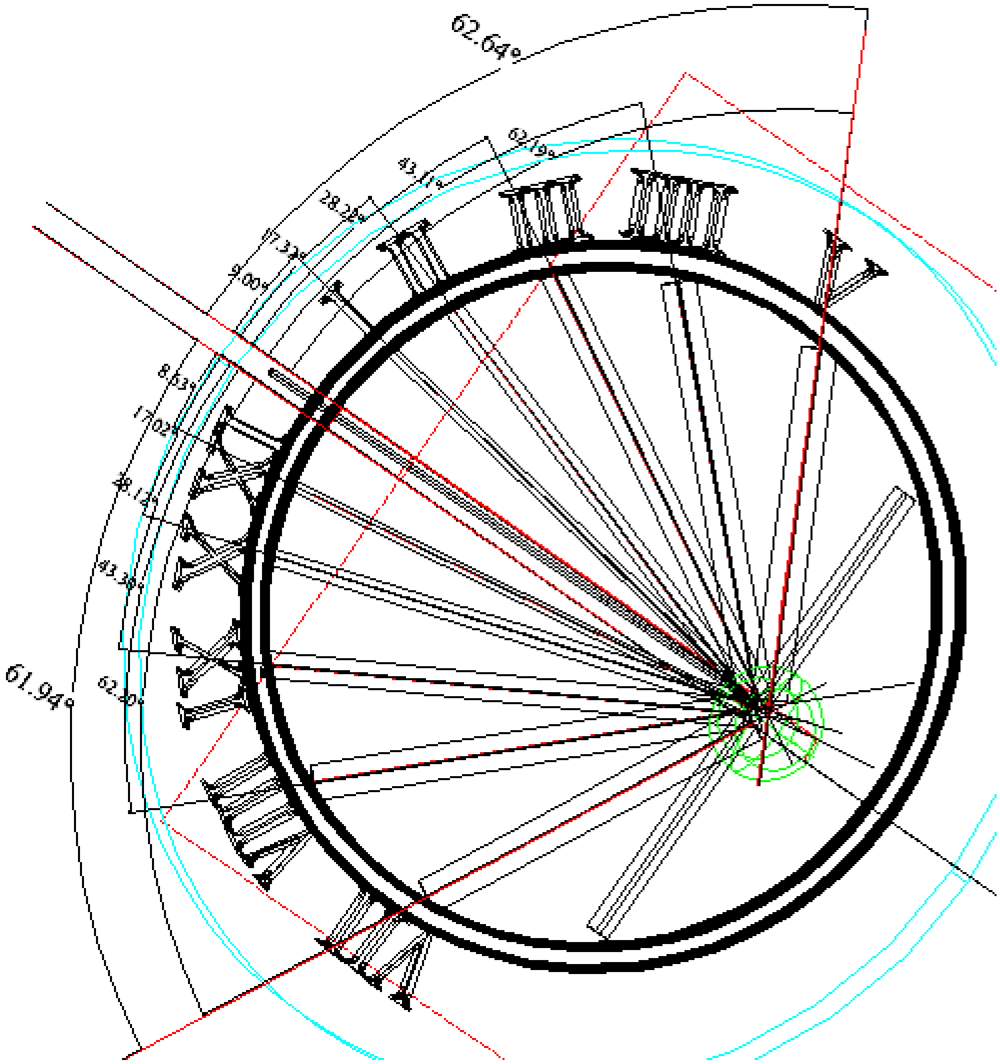
Some cloud 3d printer organizations have their own special materials.

Simple printers such as the CUBE available for around $1300 have one print head and thus multiple colors are done after the fact, requiring you to assemble the items. Some printers such as the CubeX available for around $2500 have additional options for color.

Another dial was designed with lettering and a boundary circle and then printed, see below.



3dSundialLat33.5Lng0v3circle.tcw
3dSundialLat33.5Lng0v3circle.stl



6	7	8	9	10	11	12	13	14	15	16	17	18
-90.0	-64.1	-43.7	-28.9	-17.7	-08.4	00.0	08.4	17.7	28.9	43.7	64.1	-90.0

		MEASURED		ERROR	
	Calculated	AM	PM	AM	PM
12	0				
11~1	8.4	8.63	9	0.23	0.6
10~2	17.1	17.02	17.32	0.08	0.22
9~3	28.9	28.12	28.22	0.78	0.68
8~4	43.7	43.3	43.11	0.4	0.59
7~5	64.1	61.94	62.64	2.16	1.46
6	90				

The error rates between the calculated design hour line angle and the final hour line angle were all within 1° except for 0700 and 1700 which was around 2°

FORMULA DEVELOPMENT:- SIN, COS, TAN, ASN, ACS, ATAN, SQRT

The various programming systems used to draw dial plates all use trigonometric formulae. Most languages provide for SIN, COS, TAN, and ATAN, and these are used in most dial drafting.

Very old legacy systems going back into the mists of time did not provide for floating point. Thus other ways of solving SIN, COS, and ATAN were needed. With no floating point, and size limits of variables, a table look up, or an iteration of calculations seeking a close fit had to be used.

A number of CAD systems did provide SIN, COS, and TAN. Such systems were:

TurboCAD	visual basic
DeltaCAD	Basic look alike
NANOCAD	visual basic, java script
FREECAD	Python
ProgeCAD	LISP
Blender	Python (more a modeling system than CAD)

One 2d CAD system provided a Pascal like language for macros, however documentation was sparse, and extensive research showed no SIN, COS, TAN, nor ATAN support.

Powerdraw	Pascal look alike

For Powerdraw then, the very real need for SIN, COS, TAN, and ATAN to be developed was raised.

The series for SIN, COS, and TAN were straightforward, and located in many places online, as well as in books of tables such as:-

“Mathematical Tables And Formulas” by Carmichael and Smith

However, for ATAN, two series were provided. One was useful for ranges to 45 degrees, this was somewhat limiting for sundial design, it was formula 9 in the aforementioned book. Another series was found in the same book as formula 22, and had no angular limits. These formulae were tested in Excel first:-

ATAN series formula 9 and 22.xls

And then converted into Pascal like code for Powerdraw. In Excel, the series from formula 22 worked well to about 80 degrees when it became error prone. This was due no doubt to the limits of the SCIENTIFIC variable. And after about 150 terms, more limits were exceeded. When coded in Powerdraw’s Pascal like macro language some problems arose.

Erroneous results got worse around 50 degrees, again probably a content limit for REAL or DOUBLE variables. Thus an iterative method was used for Powerdraw sundial macros, harking back to the IBM 1401 methods the author used. Further, errors happened when a formula such as:

```
tanx := sin(x) / cos(x);                // INSTEAD USE:  s=sin(x); c=cos(x); tanx:-s/c;
```

was used. Powerdraw produced erroneous results when a user function (ATAN) invoked other user functions such as SIN and COS. Thus inline code was needed.

This article discusses the Pascal like functions for SIN and COS, the derived TAN, and then the experiments for ARCSIN, ARCCOS, and ARCTAN, ending up with an iterative loop seeking a good fit.

THE SIN AND COS SERIES.

```
//////////////////////////////////////////////////////////////////////////
function sin( aSINin:Double):Double;
   /////////////////////////////////////////////////////////
   // works for all angles positive and negative -360 to 450
   var
   mysin,a: double;
   begin
      a := aSINin;
      if aSINin > 180 then begin         // 2013 12 23
          a:= aSINin-180;                // fix if angle great than 270
      end ;                              //
      // sin = x - (x**3)/3! + (x**5)/5! - (x**7)/7! + . . .
      a := a*2*3.14159/360 ;
      mysin := a
                - ((a*a*a)/(3*2))
                + ((a*a*a*a*a)/(5*4*3*2))
                - ((a*a*a*a*a*a*a)/(7*6*5*4*3*2))
                + ((a*a*a*a*a*a*a*a*a)/(9*8*7*6*5*4*3*2))
                - ((a*a*a*a*a*a*a*a*a*a*a)/(11*10*9*8*7*6*5*4*3*2))
                + ((a*a*a*a*a*a*a*a*a*a*a*a*a)/(13*12*11*10*9*8*7*6*5*4*3*2))
                - ((a*a*a*a*a*a*a*a*a*a*a*a*a*a*a)/(15*14*13*12*11*10*9*8*7*6*5*4*3*2))
                ;
      result:= mysin ;
      if aSINin > 180 then begin         // 2013 12 23
          result:= -mysin;               // fix if angle great than 270
      end ;                              //
   end;
//////////////////////////////////////////////////////////////////////////


//////////////////////////////////////////////////////////////////////////
function cos( aCOSin:Double):Double;
   /////////////////////////////////////////////////////////
   // works for all angles positive and negative -360 to 450
   var
   mycos,a: double;
   begin
      a := aCOSin;
      if aCOSin > 180 then begin         // 2013 12 23
          a:= aCOSin-180;                // fix if angle great than 270
      end ;                              //
      if aCOSin <0 then begin            // 2013 12 24
          a:= 180+acosin;                // fix negative COS
      end;
      // cos = 1 - (x**2)/2! + (x**4)/4! - (x**6)/6! + . . .
      a := a*2*3.14159/360 ;
      mycos := 1
                - ((a*a)/(2))
                + ((a*a*a*a)/(4*3*2))
                - ((a*a*a*a*a*a)/(6*5*4*3*2))
                + ((a*a*a*a*a*a*a*a)/(8*7*6*5*4*3*2))
                - ((a*a*a*a*a*a*a*a*a*a)/(10*9*8*7*6*5*4*3*2))
                + ((a*a*a*a*a*a*a*a*a*a*a*a)/(12*11*10*9*8*7*6*5*4*3*2))
                - ((a*a*a*a*a*a*a*a*a*a*a*a*a*a)/(14*13*12*11*10*9*8*7*6*5*4*3*2))
                ;
      if aCOSin > 0   then   result:= mycos ;        // a. 2013 12 16
      if aCOSin <0.01 then   result:= 1;             // b. 2013 12 19
      if aCOSin <0    then   result:= -mycos;        // c. 2013 12 19

      if aCOSin > 180 then begin                     // 2013 12 23
         if aCOSin > 0   then   result:= -mycos ;    // a. 2013 12 16
         if aCOSin <0.01 then   result:= -1;         // b. 2013 12 19
         if aCOSin <0    then   result:= mycos;      // c. 2013 12 19
      end ;                                          //
   end;
//////////////////////////////////////////////////////////////////////////
```

The above short series are accurate enough that an iterative process was not needed because the double attribute for the variable MYSIN and MYCOS was sufficient for accurate results.

THE TAN SERIES.

The common series could have been used, see immediately below, however the final function used an inline sin and cos, then divided the results. This was because of insufficient significant digits in the variables in the Pascal variant used, so at higher angles, errors were substantial.

```
function tanSeries( a:Double):Double;
   var
   mytan: double;
   begin
      // tan = x + (x**3)/3 + 2*(x**5)/15 + 17*(x**7)/315 + . . .
      a := a*2*3.14159/360 ;
      mytan := a  + ((a*a*a)/(3))
                  + (2*(a*a*a*a*a)/(15))
                  + (12*(a*a*a*a*a*a*a)/(315)) ;
      result:= mytan ;
   end;
```

The reason an inline cos and sin were used was because Powerdraw's implementation of Pascal produced erroneous results when a user function (sin, cos) were used within a user function (tan).

```
///////////////////////////////////////////////////////////////////////////
function tan( aTANin:Double):Double;
   ////////////////////////////////////////////////////////
   // works for all angles positive and negative -360 to 450
   var
   mysin,a: double;
   var
   mycos: double;
   begin
      a := aTANin*2*3.14159/360 ;
      if aTANin > 180 then begin          // 2013 12 23
          a:= (aTANin-180)*2*3.14159/360; // fix if angle great than 270
      end ;                               //
      // sin = x - (x**3)/3! + (x**5)/5! - (x**7)/7! + . . .
      mysin := a
               - ((a*a*a)/(3*2))
               + ((a*a*a*a*a)/(5*4*3*2))
               - ((a*a*a*a*a*a*a)/(7*6*5*4*3*2))
               + ((a*a*a*a*a*a*a*a*a)/(9*8*7*6*5*4*3*2))
               - ((a*a*a*a*a*a*a*a*a*a*a)/(11*10*9*8*7*6*5*4*3*2))
               + ((a*a*a*a*a*a*a*a*a*a*a*a*a)/(13*12*11*10*9*8*7*6*5*4*3*2))
               - ((a*a*a*a*a*a*a*a*a*a*a*a*a*a*a)/(15*14*13*12*11*10*9*8*7*6*5*4*3*2))
               ;
      // cos = 1 - (x**2)/2! + (x**4)/4! - (x**6)/6! + . . .
      mycos := 1
               - ((a*a)/(2))
               + ((a*a*a*a)/(4*3*2))
               - ((a*a*a*a*a*a)/(6*5*4*3*2))
               + ((a*a*a*a*a*a*a*a)/(8*7*6*5*4*3*2))
               - ((a*a*a*a*a*a*a*a*a*a)/(10*9*8*7*6*5*4*3*2))
               + ((a*a*a*a*a*a*a*a*a*a*a*a)/(12*11*10*9*8*7*6*5*4*3*2))
               - ((a*a*a*a*a*a*a*a*a*a*a*a*a*a)/(14*13*12*11*10*9*8*7*6*5*4*3*2))
               ;
      result:= mysin/mycos ;
   end;
///////////////////////////////////////////////////////////////////////////
```

The SIN, COS, and TAN work for all the angles and for positive and negative numbers. The ARCSIN, ARCCOS, and ARCTAN (asn, acs, atan) all use an iterative process generating a sin/cos/tan as appropriate until a fit is found. This was because the system used (Powerdraw's Pascal subset) did not provide variables with enough significant digits.

THE ARCSIN SERIES AND ITERATIVE PROCESS FOR ALL ANGLES IN FINAL FORM

The first function using a series fails probably due to variable content issues.

```
function asnSeries( a:Double):Double;
   // ERRONEOUS ANSWER FOR LARGER ANGLES PROBABLY DUE TO VARIABLE CONTENT LIMITS
   var    myasn: double;
   begin
      myasn := a
          + ( (1)           /(2)               ) * ( (a*a*a)                     /3  )
          + ( (1*3)         /(2*4)             ) * ( (a*a*a*a*a)                 /5  )
          + ( (1*3*5)       /(2*4*6)           ) * ( (a*a*a*a*a*a*a)             /7  )
          + ( (1*3*5*7)     /(2*4*6*8)         ) * ( (a*a*a*a*a*a*a*a*a)         /9  )
          + ( (1*3*5*7*9)   /(2*4*6*8+10)      ) * ( (a*a*a*a*a*a*a*a*a*a*a)     /11 )
      //  + ( (1*3*5*7*9*11) /(2*4*6*8+10+12)   ) * ( (a*a*a*a*a*a*a*a*a*a*a*a*a) /13 );
      result:= (myasn*360)/(2*3.14159) ;
   end;
```

The second function shown below using an iterative loop works well.

```
////////////////////////////////////////////////////////////////////////////
function asn( aASNin:Double):Double;
   /////////////////////////////////////////
   // works for positive and negative angles
   var
   myasn, myasnincr,asns1,asnnw,p,q,r: double;

   begin                              // "a" input is a tangent value
      myasn        :=  0 ;            // output result is in degrees*100  )  100 to 1
      myasnincr    :=  1;             // assuming the increment is 0.01   )  100 to 1
      result       :=  123456;        // impossible result

      p := aASNin;                    // make "p" positive
      if aASNin <0 then p := -aASNin;  // as loop assumes positive in compare logic

      while myasn < 8999  do begin     // this limit number should get to 89.9
            // problem invoking sin and cos in a user function in Powerdraw
            q  := (myasn/100)*2*3.14159/360;                              // 100 to 1
            // sin = x - (x**3)/3! + (x**5)/5! - (x**7)/7! + . . .
            asns1 := q
                - ((q*q*q)/(3*2))
                + ((q*q*q*q*q)/(5*4*3*2))
                - ((q*q*q*q*q*q*q)/(7*6*5*4*3*2))
                + ((q*q*q*q*q*q*q*q*q)/(9*8*7*6*5*4*3*2))
                - ((q*q*q*q*q*q*q*q*q*q*q)/(11*10*9*8*7*6*5*4*3*2))
                + ((q*q*q*q*q*q*q*q*q*q*q*q*q)/(13*12*11*10*9*8*7*6*5*4*3*2))
                - ((q*q*q*q*q*q*q*q*q*q*q*q*q*q*q)/(15*14*13*12*11*10*9*8*7*6*5*4*3*2))
                ;
            if asns1  >= p then begin
               r      := myasn/100 ;    //  100 to 1
               myasn  := 999999  ;      // break the loop
            end ;

            myasn := myasn + myasnincr;
      end;

      // at this point "myasn" is the angle times 100
      p := r;                          // get what we have so far
      if myasn = 0 then begin          // use variable "p" for a new purpose
            if aASNin>0 then p := 90;   // fix 0 to 90 if sin was not 0
      end;
      if myasn = 999999 then p := 90;   // fix to 90 if sin was way high
      if aASNin = 1     then p := 90;
      result:= p;                      // now finally fix the sign
      if aASNin < 0 then result:= -p;
   end;
////////////////////////////////////////////////////////////////////////////
```

THE ARCCOS ITERATIVE PROCESS FOR ANGLES UP TO 90 DEGREES IN FINAL FORM

The function shown below using an iterative loop works well.

```
////////////////////////////////////////////////////////////////////////////
function acs( aACSin:Double):Double;
   /////////////////////////////////////////////
   // works for positive and negative angles
   var                                // 2013 12 25 1710 mst fixes
   myacs, myacsincr,acsc1,acsnw,p,q,r: double;

   begin                              // "a" input is a tangent value
      myacs         :=  0 ;           // output result is in degrees*100  )  100 to 1
      myacsincr     :=  1;            // assuming the increment is 0.01   )  100 to 1
      result        :=  123456;       // impossible result

      p := aACSin;                    // make "p" positive
      if aACSin <0 then p := -aACSin;  // as loop assumes positive in compare logic
      while myacs < 8999  do begin     // this limit number should get to 89.9
            // problem invoking sin and cos in a user function in Powerdraw
            q  := (myacs/100)*2*3.14159/360;                                  // 100 to 1
            // cos = 1 - (x**2)/2! + (x**4)/4! - (x**6)/6! + . . .
            acsc1 := 1
                - ((q*q)/(2))
                + ((q*q*q*q)/(4*3*2))
                - ((q*q*q*q*q*q)/(6*5*4*3*2))
                + ((q*q*q*q*q*q*q*q)/(8*7*6*5*4*3*2))
                - ((q*q*q*q*q*q*q*q*q*q)/(10*9*8*7*6*5*4*3*2))
                + ((q*q*q*q*q*q*q*q*q*q*q*q)/(12*11*10*9*8*7*6*5*4*3*2))
                - ((q*q*q*q*q*q*q*q*q*q*q*q*q*q)/(14*13*12*11*10*9*8*7*6*5*4*3*2))
                ;
            r     := myacs/100 ;        // 2013 12 25
            if acsc1  <= p then begin
               myacs  := 998877  ;      // myacs to 99999 breaks the while loop
            end ;
            myacs := myacs + myacsincr;
      end;
      // at this point "r" is the angle - now use "p" for a new purpose
      p := r;                           // p is the angle so far
      if myacs = 998877 then  p := 90;  // fix to 90 if cos loop ended
      if myacs = 0      then  p := 90;  // fix to 90 if cos loop ended
      if myacs = 0 then begin           // fix to 90  if cos loop stopped first time
         if aACSin>=0   then  p := 90;  // ... but input was not 0
      end;
      if aACSin =0      then  p := 90;  // 2013 12 25 fix these cases
      if aACSin >0.9999 then  p :=  0;  // fix to 0 degrees
      result:= p;                       // result is now in place

      if aACSin < 0 then p:= -p;        // but finally fix the sign
   end;
////////////////////////////////////////////////////////////////////////////
```

THE ATAN SERIES FOR ANGLES UP TO 45 DEGREES

The series for angles up to a 45 degrees, page 266 formula 9 "Mathematical Tables And Formulas" by Carmichael and Smith, is:-

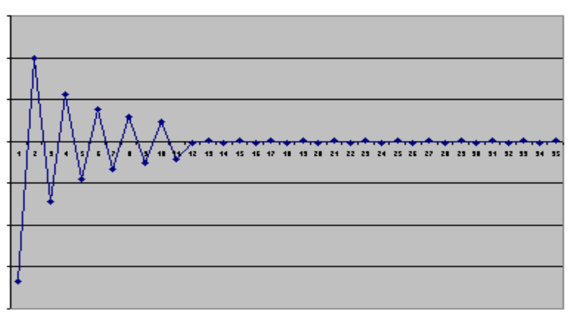
atan(z) = z – z**3/3 + z**5/5 – z**7/7 + z**9/9 – . . .

This is simple but somewhat useless for gnomonics, since angles range from -90 to +90 degrees. However, the following code produces accurate results to 45 degrees.

```
var
a,x,y,z,rad,latitudei: intger;
tanA,tanO,tanH,tangle,myangle,latitude: double;
dt2shr,dt2chr,dt2thr: double;

begin
   latitudei:= StrtoInt( inputbox('An angle','in degrees',''));
   latitude := latitudei*2*3.14159/360;
   dt2shr   :=  sin(latitude);        // This is test harness code
   dt2chr   :=  cos(latitude);        // This is test harness code
   dt2thr   :=  dt2shr/dt2chr;        // This is test harness code
   showmessage(dt2thr);               // dt2thr is the TAN of an angle
   myangle :=  (z -  z*z*z*3/3 + z*z*z*z*z*5/5 -  z*z*z*z*z*z*z*7/7 +
               z*z*z*z*z*z*z*z*z*9/9) *360 / (2*3.14159);
   showmessage(myangle);
   Refresh;
end.
```

The 45 degree limited series, in iterative form is:-

```
// "z"            is the tangent of the angle
// "myatanx"      is the resulting angle in radians
// "t1"           is the factor, top line, that alternates
// "b1"           is the factor, bottom line
// around 30 degrees, this code becomes erroneous
// and 100 iterations is not improved when going to 500
// this is thus limited by the capacity of the variables
// and changing variables from DOUBLE to REAL has no affect
t1    :=(-1)*z*z*z;
b1   :=3;
for   i:=1 to 100 do
    begin
       myatanx    := z    +   t1/b1     ;
       t1         :=  (-1)*t1*z*z;      // alternates sign also
       b1         := b1+2;
    end;
 myatanx := myatanx * 360 / (2*3.14159);
```



The alternating nature of this iterative process for this formula using the series valid for angles up to 45 degrees.

AN ATAN SERIES FOR ANGLES UP TO 90 DEGREES

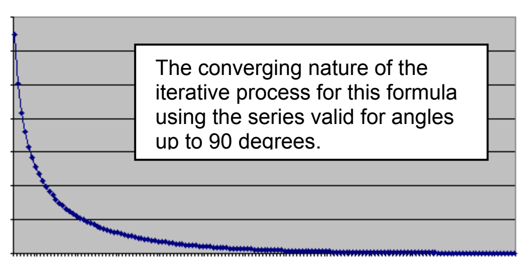
The series for angles up to a 90 degrees, page 267 formula 22 "Mathematical Tables And Formulas" by Carmichael and Smith, is:-

```
atan(x) =  (x/(1+x*x))    *    (1   +   2*x*x/(3*(1+x*x))  +
                                        2*4*((x*x/(1+x*x))*(x*x/(1+x*x)))/(3*5) + . . .
```

This is simple, works well in tests in a spreadsheet up to angles of 80 degrees, but Powerdraw produces progressively erroneous results at lower angles than in Excel. This is most likely a result of the size limits of the variables, double or real in Pascal, or scientific in Excel.

30 -> 29   40 -> 38   50 -> 48   60 -> 65   70 -> 101   (Powerdraw results)

```
var
i,a,x,y,z,rad,latitudei: intger;
tanA,tanO,tanH,tangle,myangle,latitude, dt2shr,dt2chr,dt2thr, t1,t2,b1,b2: real;

begin
   latitudei:= StrtoInt( inputbox('An angle','in degrees',''));
   latitude := latitudei;            // This is test harness code
   dt2shr  :=  sin(latitude);        // This is test harness code
   dt2chr  :=  cos(latitude);        // This is test harness code
   dt2thr  :=  dt2shr/dt2chr;        // This is test harness code
   showmessage(dt2thr);              // This is test harness code

   ////////////////////////////////////////////////////////////////////////////////
   //     page 267 formula 22 Mathematical Tables And Formulas
   ////////////////////////////////////////////////////////////////////////////////

   //  t1          2      2*4      2*4*6 ...
   //  t1inc       2      4        6
   //  b1          3      3*3      3*5*7 ...
   //  b1incr      1      2        3
   //  m1          x*x / (1 + x*x )
   //  m1incr      multiplying m1 by m1 for b1incr times
   //  atan(x) =   atanx =  (x/(1+x*x)) * (1    +    2*x*x/(3*(1+x*x))  +
   //                       2*4*((x*x/(1+x*x))*(x*x/(1+x*x)))/(3*5) + . ..

   t1      :=   2;                t1incr:=    2;   // then 4, 6, 8, then 10, etc
   b1      :=   3;                b1incr:=    2;   // then 3, 5, then 7, etc
   m1      :=   x*x / (1+x*x);    m1incr:=  0;
   atanx:= 0 ;

   for i:=1 to 100 do begin
          m1incr := m1;
          for j:=1 to i do begin
                 if i > 1 then m1incr := m1incr * m1;
          end ;
          m1incr:= m1incr * t1 / b1;

          atanx := atanx + m1incr;

          t1      := t1 * t1incr;
          t1incr := t1incr + 2;

          b1      := b1 * b1incr;
          b1incr := b1incr + 2;
   end;
   atanx := atanx  + 1;
   atanx := atanx * x / (1 + x*x);
   atanx := 360 * atanx / (2*3.14159);
```



This was not turned into a user function because of the progressive erroneous results caused by limits on the variable contents within Powerdraw.

AN ATAN ITERATIVE PROCESS FOR ANGLES UP TO 90 DEGREES IN FINAL FORM

This ATAN function uses an iterative best fit process, calculating the sin and cos and hence the tan for each increase of an angle, until a higher TAN was found.

```
///////////////////////////////////////////////////////////////////////////
function atan( aATNin:Double):Double;
   /////////////////////////////////////////
   // works for positive and negative angles
   var
   myatan, myatanincr,atans1,atanc1,atant1,atanz,atanw,q,p,r: double;

   begin                               // "a" input is a tangent value
      myatan        :=  0 ;            // output result is in degrees*100  )  100 to 1
      myatanincr    :=  1;             // assuming the increment is 0.01   )  100 to 1
      result        :=  123456;        // impossible result

      p := aATNin;                     // make "p" positive
      if aATNin <0 then p := -aATNin;  // as loop assumes positive in compare logic

      while myatan < 8999  do begin    // this limit number should get to 89.9
            // problem invoking sin and cos in a user function in Powerdraw
            q  := (myatan/100)*2*3.14159/360;                          // 100 to 1
            // sin = x - (x**3)/3! + (x**5)/5! - (x**7)/7! + . . .
            atans1 := q
                - ((q*q*q)/(3*2))
                + ((q*q*q*q*q)/(5*4*3*2))
                - ((q*q*q*q*q*q*q)/(7*6*5*4*3*2))
                + ((q*q*q*q*q*q*q*q*q)/(9*8*7*6*5*4*3*2))
                - ((q*q*q*q*q*q*q*q*q*q*q)/(11*10*9*8*7*6*5*4*3*2))
                + ((q*q*q*q*q*q*q*q*q*q*q*q*q)/(13*12*11*10*9*8*7*6*5*4*3*2))
                - ((q*q*q*q*q*q*q*q*q*q*q*q*q*q*q)/(15*14*13*12*11*10*9*8*7*6*5*4*3*2))
                ;
            // cos = 1 - (x**2)/2! + (x**4)/4! - (x**6)/6! + . . .
            atanc1 := 1
                - ((q*q)/(2))
                + ((q*q*q*q)/(4*3*2))
                - ((q*q*q*q*q*q)/(6*5*4*3*2))
                + ((q*q*q*q*q*q*q*q)/(8*7*6*5*4*3*2))
                - ((q*q*q*q*q*q*q*q*q*q)/(10*9*8*7*6*5*4*3*2))
                + ((q*q*q*q*q*q*q*q*q*q*q*q)/(12*11*10*9*8*7*6*5*4*3*2))
                - ((q*q*q*q*q*q*q*q*q*q*q*q*q*q)/(14*13*12*11*10*9*8*7*6*5*4*3*2))
                ;
            // get the resulting tangent
            atant1  := atans1/atanc1;      // still in radians
            // if we are equal or greater than, then myatan is the result
            if atant1 >= p then begin
               r      := myatan/100;       // set the answer into "r" divided by 100
               myatan := 999999  ;         // break the loop
            end ;

            myatan := myatan + myatanincr;
      end;

      // at this point "myatan" is the angle times 100
      p := r;                              // get what we have so far
      if myatan = 0 then begin             // use variable "p" for a new purpose
            if aATNin>0  then  p := 90;    // fix 0 to 90 if tan was not 0
      end;
      if myatan = 999999 then p := 90;     // fix to 90 if tan was way high
      if aATNin >573     then p := 90.0;   // fix 90 degree tangents

      result:= p;                          // now finally fix the sign
      if aATNin < 0 then result:= -p;
   end;
///////////////////////////////////////////////////////////////////////////
```

A SQUARE ROOT FUNCTION

This function returns “-1” for negative numbers, “0” for 0, “1” for 1, and works for numbers between 0 and 1, and for numbers above 1.

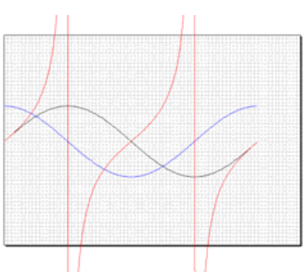
```
///////////////////////////////////////////////////////////////////////////
// sqrt (n) where n>1, n=1, n=0, n=-1, n=0.000001 to 0.999999
///////////////////////////////////////////////////////////////////////////
function sqrt( a:Double):Double;
  var
  iloop: integer;
   ilow, imid, ihigh, ioriginal, isqar, itemp:  real;
  begin
  ioriginal := a;
   ////////////////////////////////////////////////
   // sqrt 64       8      works
   // sqrt  1       1      works
   // sqrt  0       0      works
   // sqrt -nnn     -1     works
   // sqrt .64      .8     works
   iloop := 2000;                               // loop limit
   ilow  := 0;
   ihigh := ioriginal;
   imid  := 1;                                  // set a default

   if ioriginal >0 then begin
      // real numbers as not negative
      if ioriginal > 1 then begin        // THIS WORKS
         // square root of a number greater than 1
         imid := (ilow+ihigh)/2 ;               // square root so far

         while iloop > 1 do begin

            isqar := imid * imid ;              // square it

            if isqar = ioriginal then begin
               iloop :=  -1;                    // break the loop
            end ;
            if isqar < ioriginal then begin
               itemp := imid;
               imid  := (imid + ihigh) /2;
               ilow  := itemp;
            end ;
            if isqar > ioriginal then begin
               itemp := imid;
               imid  := (imid + ilow) /2;
               ihigh := itemp;
            end ;

            iloop := iloop - 1;
         end;
      end;

      // if ioriginal=1 then begin;               // THIS IS NOT BEING HIT
      if ioriginal    < 1.0000000001 then begin   // SO USE A RANGE INSTEAD
         if ioriginal > 0.9999999999 then begin   // THIS NOW WORKS
            imid := 1;
         end ;
      end;

      if ioriginal < 1 then begin                 // This is being hit

         // square root of a number greater than 0
         imid := (ilow+ihigh)/2 ;               // square root so far
         ihigh := 1 ;                           // for 0.0001 to 0.9999

         while iloop > 1 do begin

            isqar := imid * imid ;              // square it

            if isqar = ioriginal then begin
```

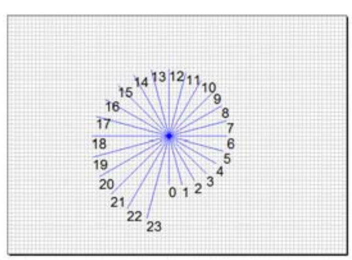
```
            iloop :=  -1;                      // break the loop
         end ;
         if isqar < ioriginal then begin
            itemp := imid;
            imid  := (imid + ihigh) /2;
            ilow  := itemp;
         end ;
         if isqar > ioriginal then begin
            itemp := imid;
            imid  := (imid + ilow) /2;
            ihigh := itemp;
         end ;

         iloop := iloop - 1;
      end;
   end;
 end;

   if ioriginal <0 then begin                  // THIS WORKS
      // imaginary numbers as negative
      imid := -1;
   end;

   // if ioriginal =0 then begin               // THIS IS NOT BEING HIT
   if ioriginal    < 0.0000000001 then begin   // SO USE A RANGE INSTEAD
      if ioriginal > -0.9999999999 then begin  // THIS WORKS
         imid := 0;
      end ;
   end;

   result:=imid;
end;
/////////////////////////////////////////////////////////////////////////////
```

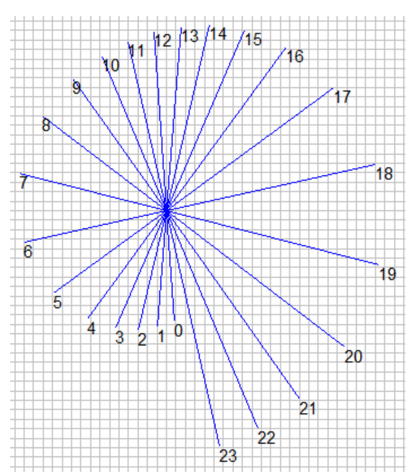
The final version of the above collected functions is on the web site and CD as:-

`functionsPowerdrawSinCosTanAsnAcsAtanSqrt.cmf`

is correct and current. And sin, cos, and tan work for negative and positive angles between -359 and +450 degrees. This “functions” macro has a test harness producing a sin, cos, and tan waveform, a spiral of radials using sin and cos, and a horizontal northern latitude dial for a given latitude with a longitude offset (Phoenix, AZ) using atan, sin and cos, see below. Again, these were developed for aa Pascal variant that did not have sufficient precision, and has a number of bugs which had to be worked around.



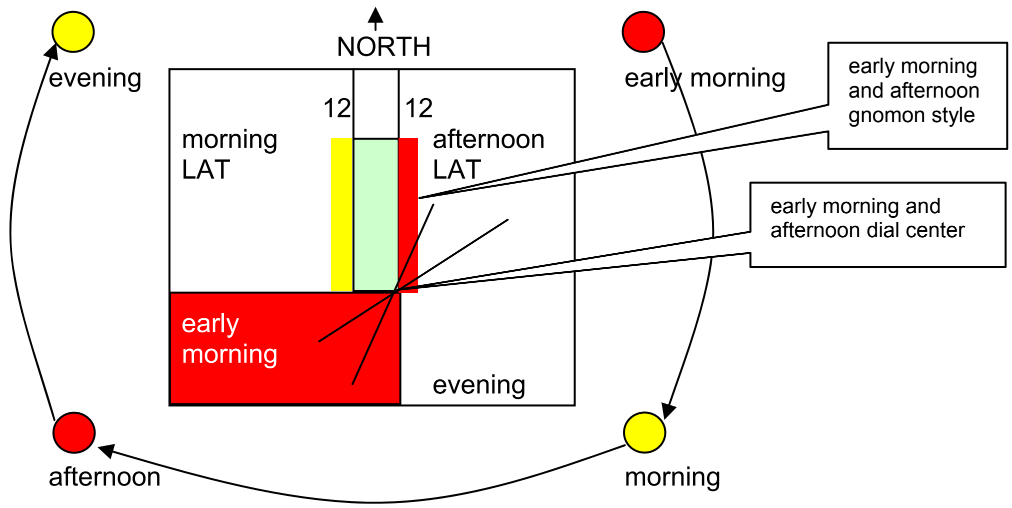




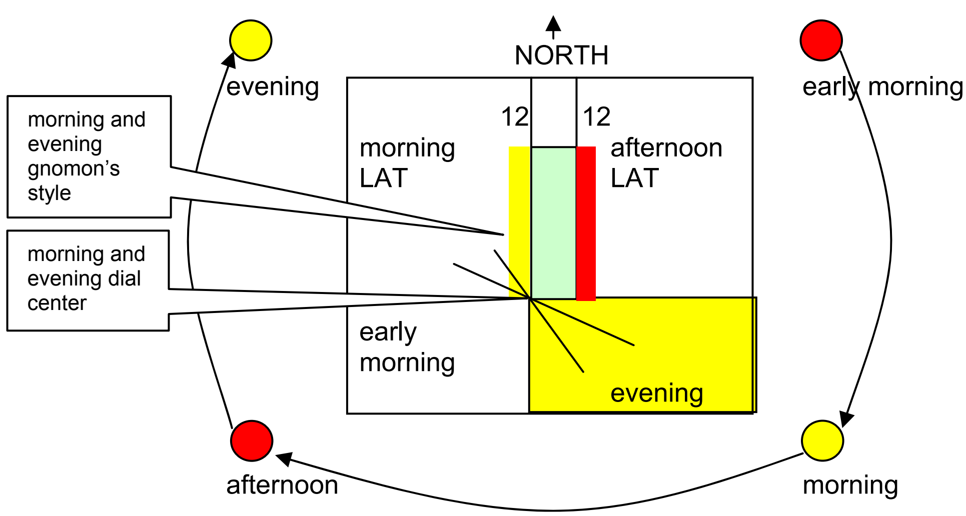
THE WIDE GNOMON and also ANALEMMAS

Fun with wide gnomons, and double S curve analemmas

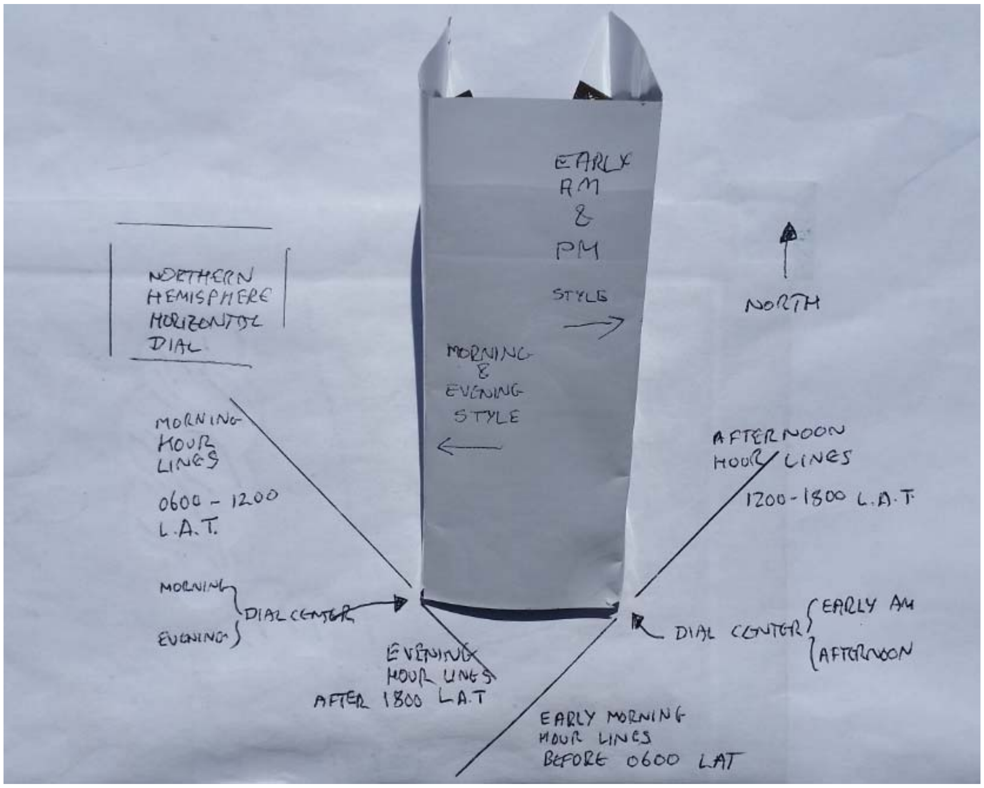
The sun moves from early morning to morning to afternoon to evening. Below is a northern hemisphere horizontal dial. Please recall that: LAT is Local Apparent Time, namely sun time and not longitude corrected nor mean time corrected (by using the EOT).



The sun has four quadrants it moves through, and the dial plate has four quadrants. The dial plate has two styles, and thus two dial centers.



A simple quick and dirty horizontal north hemisphere dial can be made, see below, to verify the above style-in-use to dial-plate-quadrant.



So far so good, not difficult.

What if there are analemmas (figure of 8 or double S) instead of hour lines.

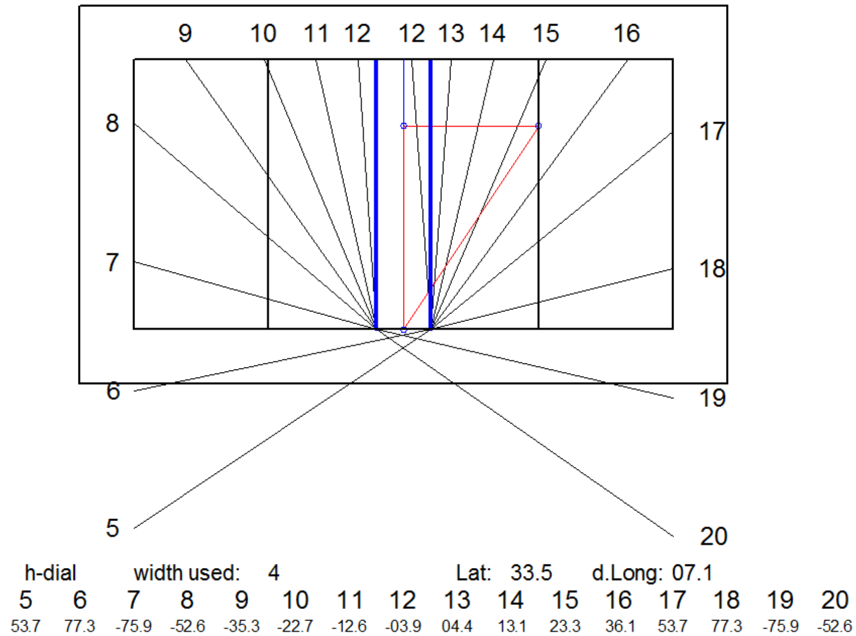
Around 6am and 6pm LAT, parts of the analemma will use one dial center, and the rest will use the other dial center. That makes it fun if you are writing a graphical program, such as the BASIC scripting of DeltaCAD.

In essence,

- if a segment of the analemma falls after 0600 LAT (normal morning) then use the relevant Northwest quadrant, SouthWest dial center and West style
- if a segment of the analemma falls before 0600 LAT (early morning) then use the relevant SouthWest quadrant, Southeast dial center, and East style
- if a segment of the analemma falls before 1800 LAT(afternoon) then use the relevant Northeast quadrant, SouthEast dial center and East style
- if a segment of the analemma falls after 1800 LAT (evening) then use the Southeast quadrant, SouthWest dial center and West style

HORIZONTAL DIAL WITH A GNOMON OF SIGNIFICANT WIDTH

The wide gnomon means, as we have seen, two styles and thus two dial centers. The dial plate below is choice 10 of the DeltaCAD horizontal dial macro.



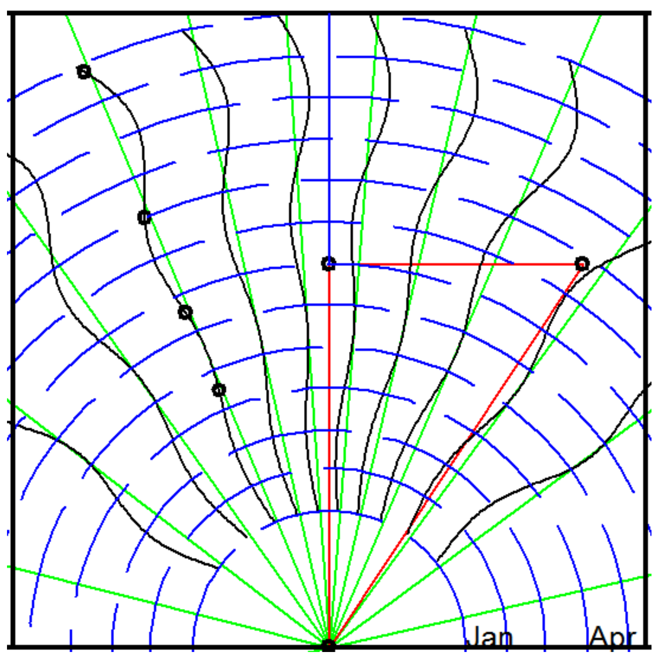
THE SPIDER DIAL, BUT WITH A NORMAL GNOMON OF INSIGNIFICANT WIDTH

Here is a horizontal dial for the northern hemisphere.

The hour curves were drawn as segments of the year, two days at a time, but considered not only longitude, but the Equation of time also; the double S analemma.

The analemma could have been a figure of 8, where the nodus of the style shows clock time. But in this case, the analemma was a double "S" shape, where the intersection of the style's shadow with a date circle indicates the time.

The logic is very simple if a thin gnomon is employed, and analemmas shown between 070 to 1700 local apparent time (LAT).



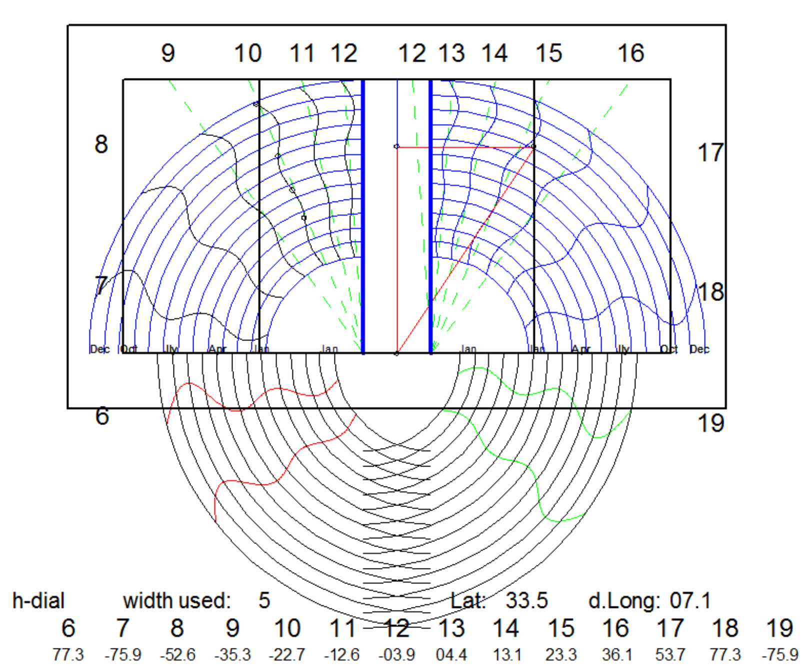
The above spider dial, or double-S-analemma dial, does not show the analemma curves around 0600 and 1800 LAT (local apparent time).

```
' ----------------------------------------------------------------------
' draw analemma
' ----------------------------------------------------------------------

dl          = lng-ref                ' longitude diff from meridian, + is west
h1          = 5                      ' 5  is 2000 hour line
h2          = 20                     ' 20 is 0500 hour line

degtorad    = 2*3.14159/360
radtodeg    = 360 / (2*3.1416)

dcSetLineParms dcBLACK ,dcSOLID,dcTHIN
For hr = h1 to h2 step 1

   ' unless 12 noon retain this sign for all x in an hour
   signOfX = 0

   For jd = 1  To 365 Step 2

       ' *** GET EOT ***
       eot  =  (7.5  * Sin(RAD(jd-5))-10.2*Sin(RAD(1.93*(jd-80)))+
                           0.5*Sin(RAD(1.5*(jd-62))) - 0.3 )/60

       '  hr   = hour
      '  jd   = day
       '  eot  = eot in hours

       ' for the hour (hr) calculate the hour line angle (h)
       h = deg(Atn(Sin(rad(lat))*Tan(rad(((hr+eot)*15) +(ref-lng)))))

       y = (jd+dispJD) * Sin(rad(h+90)) * scaleJD
       x = (jd+dispJD) * Cos(rad(h+90)) * scaleJD
       ' NOTE this uses Sin and Cos for hour line (analemma), whereas the faint
       ' hour lines use Tan to constrain the hour line in a box, hence why the
       ' EOT=0 analemma point does not perfectly match the faint hour lines
       ' that are box constrained.


       ' ALSO why this method cant do the analemma below 90 degrees to North|South

       If   jd = 1  Then
               lastx = x
               lasty = y
               frstx = x
               frsty = y
       Else
               If Abs(y) < 1 Then
                  dcCreateLine    lastx,lasty, x,y
               End If
               lastx = x
               lasty = y
       End If

       dcSetTextParms dcBLACK,"Ariel","Bold",0, 4,   8,0,0
       If jd =(31+28+31+15)                    And (hr = 15) Then
          dcCreateCircle x, y, 0.01            ' "4/15"
       End If
       If jd =(31+28+31+30+31+16)              And (hr = 15) Then
          dcCreateCircle x, y, 0.01            ' "6/15"
                                               ' [used +16 as doing JD STEP 2
       End If
       If jd =(0+31+28+31+30+31+30+31+31+0)    And (hr = 15) Then
          dcCreateCircle x, y, 0.01            ' "9/1"
       End If
       If jd =(0+31+28+31+30+31+30+31+31+30+31+30+25) And (hr = 15) Then
          dcCreateCircle x, y, 0.01            ' "12/25"
       End If

   Next jd
   'dcCreateLine    lastx,lasty, frstx,frsty

Next hr
```

```
' ----------------------------------------------------------------------
' draw analemma JD circles for each month
' ----------------------------------------------------------------------

dcSetCircleParms  dcBLUE, dcCUTTING, dcTHIN
dcCreateCircle 0,0,(0+dispJD)*scaleJD                                   ' January 1
dcCreateCircle 0,0,(0+31+dispJD)*scaleJD                                ' February 1
dcCreateCircle 0,0,(0+31+28+dispJD)*scaleJD                             ' March 1
                                                                  ' ignore leap year
dcCreateCircle 0,0,(0+31+28+31+dispJD)*scaleJD                          ' April
dcCreateCircle 0,0,(0+31+28+31+30+dispJD)*scaleJD                       ' May
dcCreateCircle 0,0,(0+31+28+31+30+31+dispJD)*scaleJD                    ' June
dcCreateCircle 0,0,(0+31+28+31+30+31+30+dispJD)*scaleJD                 ' July
dcCreateCircle 0,0,(0+31+28+31+30+31+30+31+dispJD)*scaleJD              ' August
dcCreateCircle 0,0,(0+31+28+31+30+31+30+31+31+dispJD)*scaleJD           ' September
dcCreateCircle 0,0,(0+31+28+31+30+31+30+31+31+30+dispJD)*scaleJD        ' October
dcCreateCircle 0,0,(0+31+28+31+30+31+30+31+31+30+31+dispJD)*scaleJD     ' November
dcCreateCircle 0,0,(0+31+28+31+30+31+30+31+31+30+31+30+dispJD)*scaleJD  ' December
dcCreateCircle 0,0,(0+31+28+31+30+31+30+31+31+30+31+30+31+dispJD)*scaleJD ' end of yr

dcCreateText   (0+dispJD)*scaleJD,                           0 ,  0.1,  "Jan  1"
dcCreateText   (0+31+28+31+dispJD)*scaleJD,                  0 ,  0.1,  "Apr  1"
dcCreateText   (0+31+28+31+30+31+30+dispJD)*scaleJD,         0 ,  0.1,  "Jly  1"
dcCreateText   (0+31+28+31+30+31+30+31+31+30+dispJD)*scaleJD, 0 ,  0.1,  "Oct  1"
dcCreateText   (365+dispJD)*scaleJD,                         0 ,  0.1,  "end of yr"
```

The above is code for DeltaCAD, and assumes a thin gnomon, and does not address analemmas until at least an hour after 0600 local apparent time and an hour before 1800 local apparent time. This is choice 9 of the DeltaCAD horizontal dial macro.

For a wide gnomon, the hour lines must use the appropriate dial center and style, and for analemmas spanning 0600 and 1800 local apparent time, the segments must asses whether they are above the east west line, or below it, and again use the appropriate style, and dial center. And additionally, the calendar or date circles have to be adjusted as well because they are centered around the dial center, and now there are two centers.



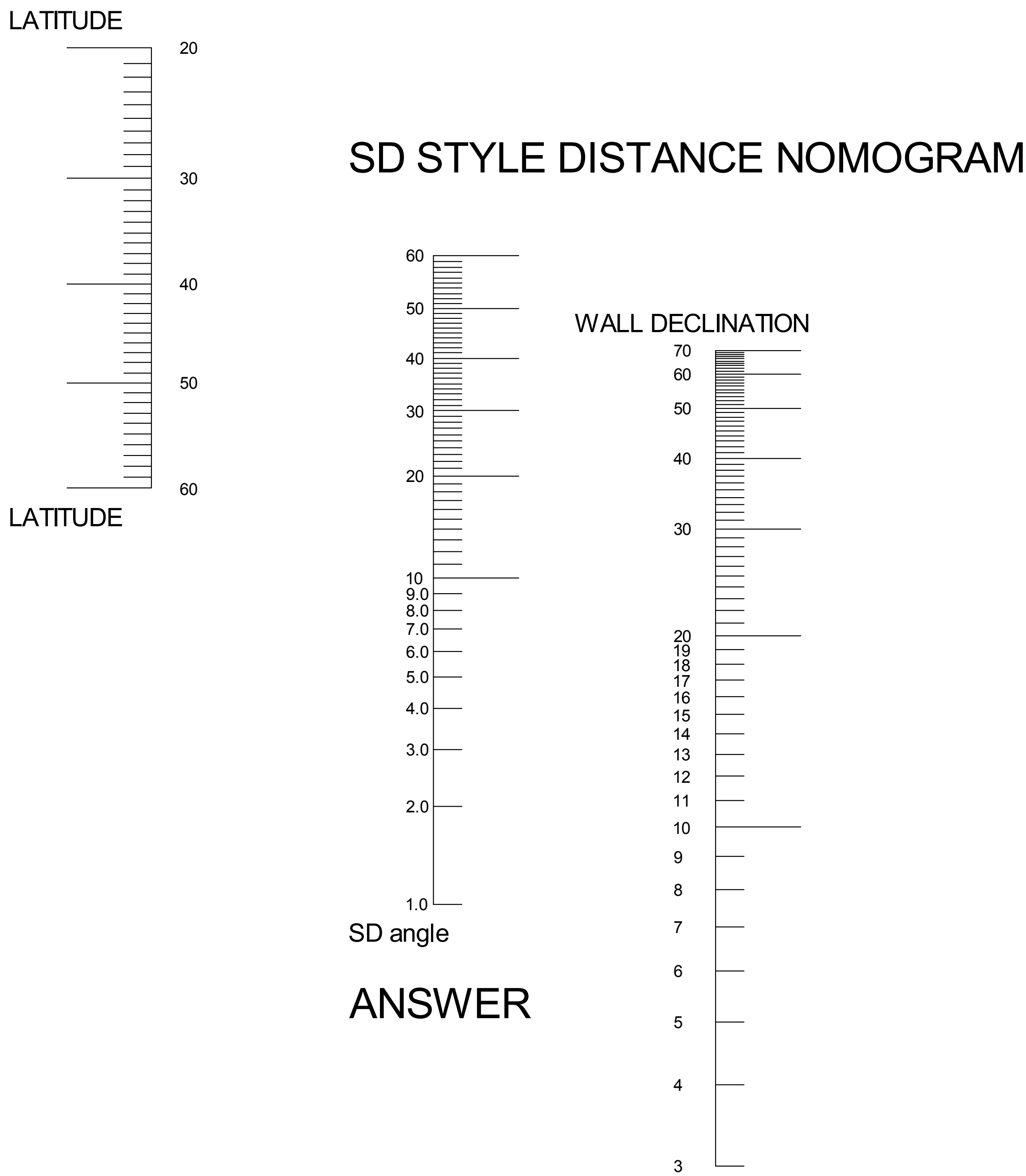
The logic is more involved than for the special case of a thin gnomon.

The complex issue arises when there is an analemma whose elements span the east west horizontal line. At that time, which is within 16 minutes of 0600 or 1800 LAT (Local Apparent Time) special considerations must be made. However, the transition of the shadow from one style to the other, may make accurate reading somewhat obscured.

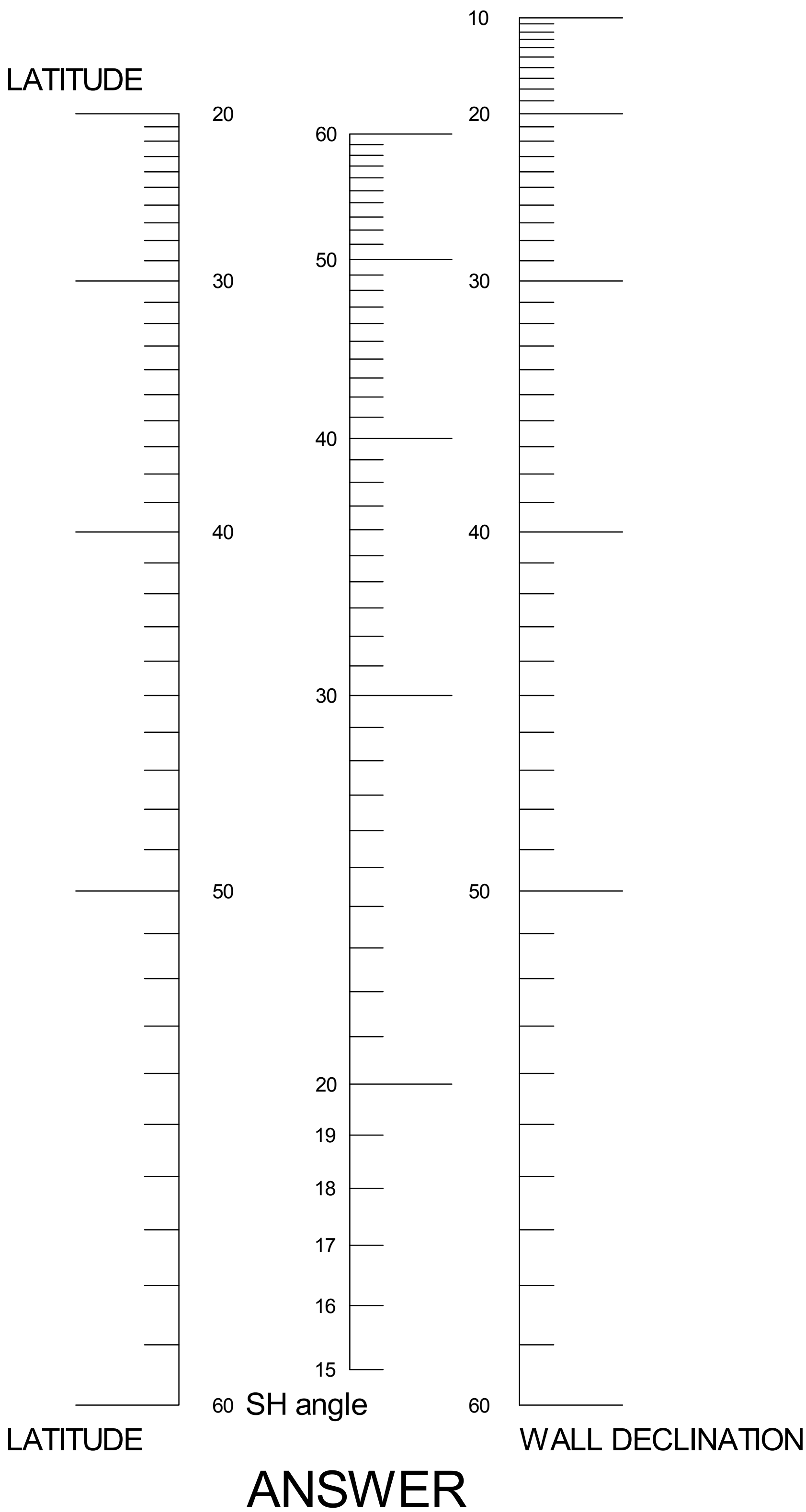
The above dial plate was produced by a DeltaCAD macro with additional logic over and above the code in the last couple of pages.

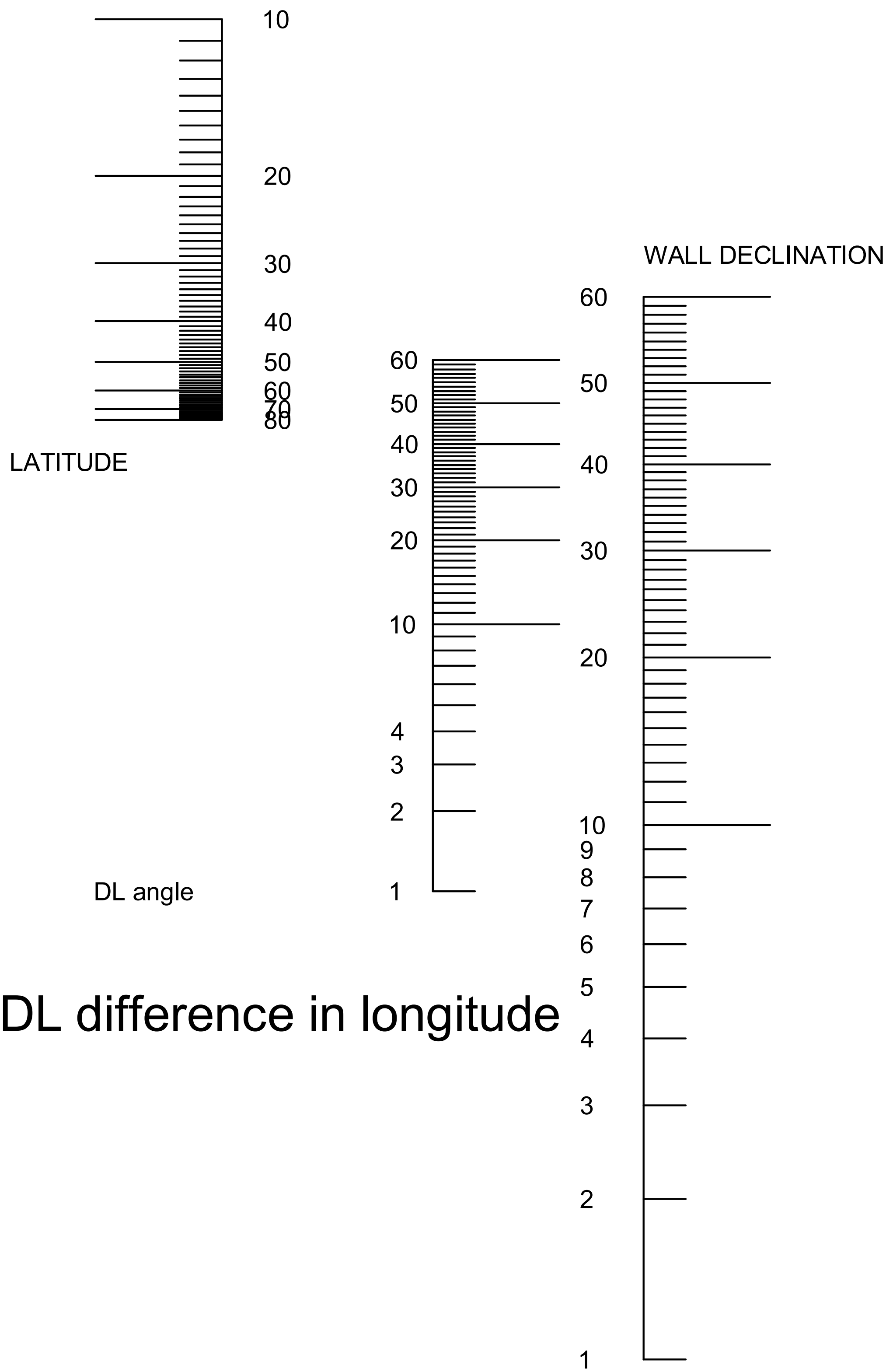
APPENDICES ~ SD, SH, and DL

The following three nomograms can be used to construct a vertical decliner, see chapter 31 and the use of “DL”. In essence, to construct a vertical decliner, calculate the SD, SH, and DL for the vertical decliner which these three nomograms do. Then design a horizontal dial for a latitude of SH, slide its local apparent noon line onto the SD line of the vertical decliner’s dial plate. And that horizontal dial’s hour lines are altered by an assumed longitude of “DL” with a legal meridian of 0. Remember that horizontal and vertical dial shadows rotate in opposite directions, so the sense of the surrogate dial’s hours needs reversing, chapter 13. **NOTE:** a vertical dial of latitude “90 – SH” can be used in place of a horizontal dial, when the sense of the hours would not need reversing.



SH STYLE HEIGHT NOMOGRAM





DL difference in longitude

FOR CROSS CHECKING V-DEC DIAL “**SD**” VALUES

VERTICAL DECLINER "SD" style distance offset for latitudes 30 to 60, declinations 1 to 80

A5.4	LATITUDE							
DECL	30	32	34	36	38	40	42	44
1	1.73	1.60	1.48	1.38	1.28	1.19	1.11	1.04
2	3.46	3.20	2.96	2.75	2.56	2.38	2.22	2.07
4	6.89	6.37	5.90	5.48	5.10	4.75	4.43	4.13
6	10.26	9.50	8.81	8.19	7.62	7.10	6.62	6.18
8	13.55	12.56	11.66	10.84	10.10	9.42	8.79	8.20
10	16.74	15.53	14.44	13.44	12.53	11.69	10.92	10.19
15	24.15	22.50	20.99	19.61	18.33	17.14	16.04	15.00
20	30.64	28.69	26.89	25.21	23.64	22.18	20.80	19.50
25	36.20	34.07	32.07	30.19	28.41	26.73	25.14	23.64
30	40.89	38.67	36.55	34.54	32.62	30.79	29.04	27.37
35	44.81	42.55	40.38	38.29	36.28	34.36	32.50	30.71
40	48.07	45.81	43.62	41.50	39.45	37.45	35.52	33.65
45	50.77	48.53	46.35	44.22	42.15	40.12	38.14	36.21
50	53.00	50.80	48.64	46.52	44.44	42.39	40.39	38.42
55	54.82	52.66	50.53	48.43	46.36	44.31	42.29	40.31
60	56.31	54.19	52.09	50.01	47.94	45.90	43.89	41.89
65	57.50	55.42	53.34	51.28	49.24	47.21	45.19	43.18
70	58.43	56.38	54.33	52.29	50.26	48.24	46.22	44.22
75	59.13	57.10	55.07	53.05	51.03	49.02	47.01	45.01
80	59.62	57.60	55.59	53.58	51.57	49.57	47.56	45.56

	46	48	50	52	54	56	58	60
1	0.97	0.90	0.84	0.78	0.73	0.67	0.62	0.58
2	1.93	1.80	1.68	1.56	1.45	1.35	1.25	1.15
4	3.85	3.59	3.35	3.12	2.90	2.69	2.50	2.31
6	5.76	5.38	5.01	4.67	4.34	4.03	3.74	3.45
8	7.65	7.14	6.66	6.21	5.77	5.36	4.97	4.59
10	9.52	8.89	8.29	7.73	7.19	6.68	6.19	5.73
15	14.03	13.12	12.25	11.43	10.65	9.90	9.19	8.50
20	18.28	17.12	16.01	14.96	13.95	12.99	12.06	11.17
25	22.20	20.83	19.53	18.27	17.07	15.91	14.79	13.71
30	25.77	24.24	22.76	21.34	19.96	18.64	17.35	16.10
35	28.98	27.31	25.70	24.14	22.62	21.15	19.72	18.32
40	31.83	30.06	28.34	26.67	25.03	23.44	21.88	20.36
45	34.33	32.48	30.68	28.92	27.19	25.50	23.84	22.21
50	36.49	34.60	32.73	30.90	29.10	27.33	25.58	23.86
55	38.35	36.41	34.50	32.62	30.76	28.92	27.11	25.31
60	39.91	37.95	36.01	34.08	32.18	30.29	28.42	26.57
65	41.19	39.22	37.25	35.30	33.36	31.44	29.52	27.62
70	42.22	40.23	38.26	36.28	34.32	32.37	30.42	28.48
75	43.01	41.01	39.03	37.04	35.06	33.09	31.11	29.15
80	43.56	41.56	39.57	37.58	35.58	33.59	31.61	29.62

This table is an aid to cross check more detailed work.

FOR CROSS CHECKING V-DEC DIAL “**SH**” VALUES

VERTICAL DECLINER "SH" style height for latitudes 30 to 60, declinations 1 to 80

A5.5	LATITUDE							
DECL	30	32	34	36	38	40	42	44
1	59.98	57.99	55.99	53.99	51.99	49.99	47.99	45.99
2	59.94	57.94	55.95	53.95	51.96	49.96	47.96	45.96
4	59.76	57.78	55.79	53.81	51.82	49.83	47.85	45.86
6	59.46	57.50	55.54	53.57	51.60	49.63	47.65	45.68
8	59.05	57.12	55.18	53.24	51.29	49.34	47.38	45.43
10	58.53	56.63	54.73	52.82	50.90	48.97	47.04	45.11
15	56.77	55.00	53.21	51.39	49.57	47.73	45.88	44.01
20	54.47	52.84	51.17	49.48	47.77	46.04	44.29	42.53
25	51.71	50.23	48.71	47.16	45.58	43.97	42.34	40.69
30	48.59	47.26	45.89	44.48	43.03	41.56	40.06	38.53
35	45.19	44.00	42.77	41.51	40.20	38.87	37.50	36.10
40	41.56	40.51	39.43	38.30	37.13	35.93	34.70	33.44
45	37.76	36.85	35.89	34.89	33.86	32.80	31.70	30.57
50	33.83	33.03	32.20	31.33	30.43	29.50	28.53	27.54
55	29.78	29.11	28.39	27.65	26.87	26.06	25.23	24.37
60	25.66	25.09	24.49	23.86	23.20	22.52	21.81	21.08
65	21.47	21.00	20.51	19.99	19.45	18.89	18.30	17.70
70	17.23	16.86	16.47	16.06	15.64	15.19	14.72	14.24
75	12.95	12.68	12.39	12.09	11.77	11.44	11.09	10.73
80	8.65	8.47	8.28	8.08	7.86	7.64	7.41	7.18

	46	48	50	52	54	56	58	60
1	43.99	41.99	39.99	37.99	35.99	33.99	31.99	29.99
2	43.97	41.97	39.97	37.97	35.97	33.98	31.98	29.98
4	43.87	41.87	39.88	37.89	35.90	33.91	31.91	29.92
6	43.70	41.72	39.74	37.76	35.77	33.79	31.80	29.82
8	43.46	41.50	39.53	37.57	35.60	33.62	31.65	29.68
10	43.17	41.22	39.27	37.32	35.37	33.41	31.46	29.50
15	42.14	40.27	38.38	36.49	34.59	32.69	30.79	28.88
20	40.75	38.96	37.16	35.35	33.53	31.70	29.87	28.02
25	39.02	37.33	35.63	33.92	32.19	30.45	28.70	26.95
30	36.98	35.41	33.83	32.22	30.60	28.97	27.32	25.66
35	34.68	33.24	31.77	30.29	28.78	27.26	25.73	24.18
40	32.15	30.84	29.50	28.14	26.76	25.36	23.95	22.52
45	29.42	28.24	27.03	25.81	24.56	23.29	22.01	20.70
50	26.52	25.47	24.40	23.31	22.20	21.07	19.91	18.75
55	23.48	22.57	21.63	20.68	19.70	18.71	17.69	16.67
60	20.32	19.55	18.75	17.93	17.09	16.24	15.36	14.48
65	17.07	16.43	15.76	15.08	14.38	13.67	12.94	12.20
70	13.74	13.23	12.70	12.16	11.60	11.03	10.44	9.85
75	10.36	9.97	9.58	9.17	8.75	8.32	7.88	7.44
80	6.93	6.67	6.41	6.14	5.86	5.57	5.28	4.98

This table is an aid to cross check more detailed work.

FOR CROSS CHECKING V-DEC DIAL “**DL**” VALUES

VERTICAL DECLINER "DL" difference in longitude for latitudes 30 to 60, declinations 1 to 80

A5.3	LATITUDE							
DECL	30	32	34	36	38	40	42	44
1	2.00	1.89	1.79	1.70	1.62	1.56	1.49	1.44
2	4.00	3.77	3.57	3.40	3.25	3.11	2.99	2.88
4	7.96	7.52	7.13	6.78	6.48	6.21	5.97	5.75
6	11.87	11.22	10.64	10.14	9.69	9.29	8.93	8.60
8	15.70	14.85	14.11	13.45	12.86	12.33	11.86	11.44
10	19.43	18.40	17.50	16.70	15.98	15.34	14.76	14.24
15	28.19	26.82	25.60	24.51	23.52	22.63	21.82	21.09
20	36.05	34.48	33.06	31.77	30.59	29.52	28.54	27.65
25	43.00	41.35	39.82	38.43	37.14	35.96	34.87	33.87
30	49.11	47.45	45.92	44.49	43.16	41.93	40.79	39.73
35	54.47	52.88	51.39	49.99	48.68	47.45	46.30	45.23
40	59.21	57.73	56.32	54.99	53.73	52.55	51.43	50.38
45	63.43	62.08	60.79	59.55	58.38	57.27	56.21	55.21
50	67.24	66.03	64.86	63.75	62.68	61.66	60.69	59.76
55	70.70	69.64	68.62	67.63	66.68	65.77	64.90	64.06
60	73.90	72.99	72.11	71.25	70.43	69.64	68.88	68.15
65	76.88	76.12	75.39	74.67	73.98	73.31	72.67	72.05
70	79.69	79.08	78.50	77.92	77.37	76.83	76.31	75.81
75	82.37	81.92	81.48	81.05	80.63	80.23	79.84	79.46
80	84.96	84.66	84.37	84.08	83.80	83.53	83.27	83.02

	46	48	50	52	54	56	58	60
1	1.39	1.35	1.31	1.27	1.24	1.21	1.18	1.15
2	2.78	2.69	2.61	2.54	2.47	2.41	2.36	2.31
4	5.55	5.38	5.22	5.07	4.94	4.82	4.71	4.62
6	8.31	8.05	7.81	7.60	7.40	7.23	7.07	6.92
8	11.05	10.71	10.40	10.11	9.85	9.62	9.41	9.22
10	13.77	13.35	12.96	12.61	12.30	12.01	11.75	11.51
15	20.43	19.83	19.28	18.78	18.33	17.91	17.53	17.19
20	26.84	26.09	25.41	24.79	24.22	23.70	23.23	22.80
25	32.95	32.11	31.33	30.62	29.96	29.36	28.80	28.30
30	38.75	37.84	37.00	36.23	35.51	34.85	34.25	33.69
35	44.23	43.30	42.43	41.62	40.88	40.18	39.55	38.96
40	49.39	48.47	47.61	46.80	46.05	45.35	44.70	44.10
45	54.27	53.38	52.55	51.76	51.03	50.34	49.70	49.11
50	58.88	58.05	57.27	56.53	55.83	55.18	54.56	53.99
55	63.27	62.51	61.79	61.11	60.47	59.86	59.30	58.77
60	67.45	66.78	66.14	65.54	64.96	64.42	63.91	63.43
65	71.46	70.89	70.34	69.82	69.33	68.86	68.42	68.01
70	75.33	74.86	74.42	74.00	73.59	73.21	72.85	72.50
75	79.09	78.74	78.40	78.08	77.77	77.48	77.20	76.94
80	82.77	82.53	82.31	82.09	81.88	81.68	81.50	81.32

This table is an aid to cross check more detailed work. Modify this with the vertical decliner's longitude difference from the legal meridian. Chapter 24 discusses this.

AVAILABLE PROGRAMMING SYSTEMS AND LANGUAGES

NOTE: The book "Programming Shadows" is strongly recommended since it takes you through the installation, odd quirks, and sample programs. It is available on:-

www.illustratingshadows.com

SPREADSHEETS

Spreadsheets are software programs that allow data to be stored in individual cells, or in columns, or in rows, and for manipulation of that data to take place.

Microsoft	Excel is a paid product
Open Office	a free package that is mostly compatible with Microsoft Office http://www.openoffice.org/
Kingsoft	a free and an upgraded version which is Excel compatible. also have a version for the Android smartphone. www.kingsoftstore.com/
e-droid-cell-lite	a free and an upgraded version which is Excel compatible designed for the Android smartphone. www.androidblip.com/android-apps/e-droid-cell-light-spreadsheet-42661.html

Most of the systems here have been tested on Windows XP, Vista, Windows 7, and Windows 8 as well as on 32 and 64 bit systems.

CAD SYSTEMS

DeltaCAD	the standard 2d CAD system used by diallists www.deltacad.com
NanoCAD	www.nanocad.com
FreeCAD	http://www.freecadweb.org/
ProgeCAD	http://www.progesoft.com/en/products/progecad-smart/
TurboCAD	http://www.turbocad.com/
Powerdraw:	http://www.powerdraw.software.informer.com
openScad	www.openscad.org
openJScad	openjscad.org
Blender	www.blender.org (a free modeling system using Python, not a true CAD system)

SCIENTIFIC SYSTEMS

Euler	http://euler.rene-grothmann.de	Scientific
Octave	http://www.gnu.org/software/octave/	" " "
SciLAB	http://www.scilab.org/	" " "

PROGRAMMING LANGUAGES

ADA	http://www.libre.adacore.com	
ALGOL	http://www.xs4all.nl/~jmvdveer/algol.html	
C and C++ and IDE: There is a free C or C++ compiler available		
	http://www.windows8downloads.com/win8-dev-c--wdoxnrth/	
	http://sourceforge.net/projects/orwelldevcpp/?source=dlp	
C#	http://msdn.microsoft.com/en-us/vstudio/hh341490.aspx	
PERL and IDE	http://www.activestate.com/activeperl	
	http://open-perl-ide.sourceforge.net/'	
FORTRAN	A free FORTRAN compiler/ linker is available at:-	
	http://gcc.gnu.org/fortran/	
	http://gcc.gnu.org/wiki/GFortran	notes about it
	http://gcc.gnu.org/wiki/GFortranBinaries	the compiler download link
PASCAL	www.bloodshed.net/devpascal.html	8mb, excellent
	www.freepascal.org/download.var	larger, less aesthetic
LAZARUS	www.lazarus.freepascal.org/	PASCAL IDE and GUI
COBOL	http://opencobol.org	not tested
vBASIC	Microsoft Visual Studio	Envelop is no more
JavScript	incorporated into most browsers	
JAVA	http://www.netbeans.info/downloads/index.php	
	http://java.sun.com/javase/downloads/index.jsp	
Python	http://www.python.org/	pyScripter inoperative

IBM MAINFRAME LANGUAGES WITH SIMULATORS

Autocoder	www.illustratingshadows.com	IBM 1401, Autocoder and SPS
Assembler	www.illustratingshadows.com	IBM 360
FORTRAN II	www.illustratingshadows.com	IBM 7094

NOTE: The book “programmingShadows.pdf” is strongly recommended since it takes you through the installation, odd quirks, and sample programs. It is available on:-

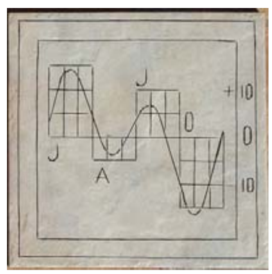
www.illustratingshadows.com

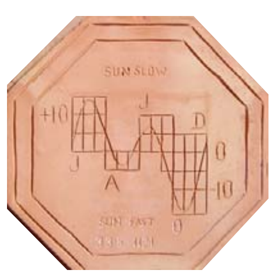
NOTE: Web sites come and go and change. The web addresses above were valid in February of 2013, however, they may change over time.

Illustrating Time's Shadow

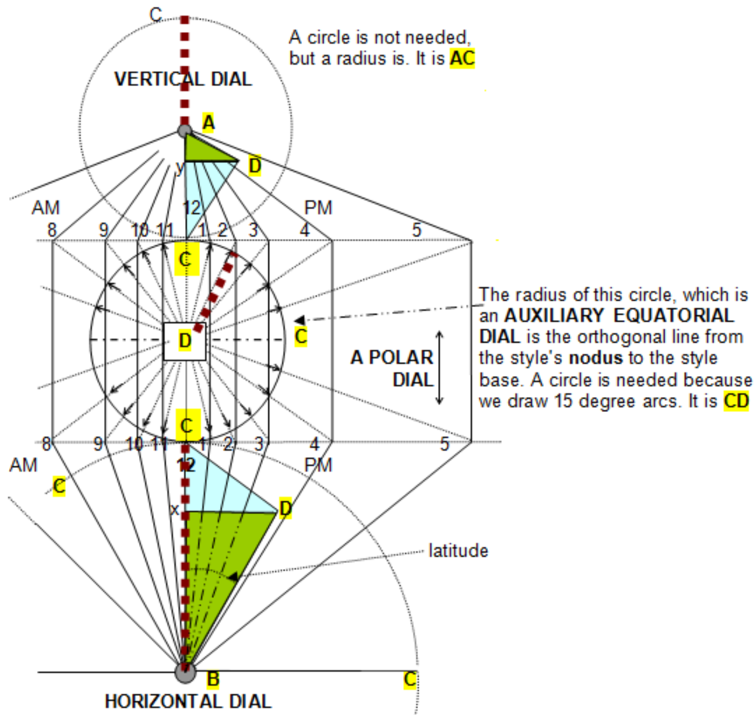
If your book's print date is later than the correction dates in the files below, then your book is updated. The website book is always up to date, you may download it any time. The QR code to the right takes you to the general reference page for all updates. Some books have their own QR code for each printing, so they have their own update page to bring clarifications into focus. All such pages refer to:-

This one common update file: **updatesAndLatestCorrections.pdf**





THIS PAGE LEFT BLANK INTENTIONALLY



The Supplements ~ Supplemental Shadows

This book addresses small indoor sundials of wood, glass, and PVC, as well as outside garden dials of glass, clay, tile, and common building materials. Less common dial features such as the inclined decliner and calendar or declination curves, are covered, as well as the astrolabe, other altitude dials and azimuth time keepers. This book uses empirical, geometric, trigonometric, CAD (computer aided design) both 2d and 3d, spreadsheet, procedural programming, tabular methods, and other techniques. Tables are provided.

The supplements are really items that are a cross between what can be in appendices and what can be in the main book. but they are more. They may also contain any corrections.

Simon Wheaton-Smith
ISBN 978-0-9960026-1-5
Library of Congress Control Number: 2014904840
ISBN 978-0-9960026-1-5
9780996002615