STATISTICAL ACCURACY OF GEOMETRIC CONSTRUCTIONS

SEPPO MUSTONEN

Abstract. In this expository compendium, a new approach for assessing complexity of classical geometric constructions is presented.

Everyone who has experience of making geometric constructions in practice knows how much attention must be paid to a careful placement of the compass and the straightedge in each step of the construction in order to achieve as accurate results as possible.

The accuracy of these placements is now described by a simple statistical model and the accuracy of the entire construction is estimated on this basis. Then it is natural to consider the accuracy of the construction as a measure of its complexity. This measure is expected to give better possibilities for comparing complexities of constructions than the characteristics of Lemoine's geometrography.

Our approach is mainly computational. Although the error distribution of placements is defined precisely, the error distributions related to entire constructions are so complicated that the only way is to use Monte Carlo simulation for estimating essential statistics.

The constructions are described by a special code in Survo which is a general environment for statistical computing and related areas. A new GEOM program module of Survo is used for creating constructions both in accurate and randomized forms. Graphs based on these constructions and summaries of simulations are accomplished by standard tools of Survo.

As applications, various constructions for a regular pentagon and for approximate circle squaring are presented. Especially in the latter task, this statistical approach gives a new possibility for comparison of various approximations. It is shown how the goodness of each construction depends fundamentally on the initial accuracy.

1. Introduction

The simplicity of geometric constructions is traditionally measured by the means of *geometrography* devised by Lemoine [10], [2], [3], [6]. Geometrography seeks to reduce the number of basic elementary operations needed for construction and the allowed operations are

S_1 Place a straightedge's graph edge through a given point,
S_2 Draw a straight line,
C_1 Place a point of a compass on a given point,
C_2 Place a point of a compass on an indeterminate point on a line,
C_3 Draw a circle.

If the number of the above operations are denoted by m_1, m_2, n_1, n_2, and n_3, respectively, then Lemoine's *simplicity* is $m_1 + m_2 + n_1 + n_2 + n_3$ and the *symbol* (of the construction) is $m_1S_1 + m_2S_2 + n_1C_1 + n_2C_2 + n_3C_3$.

Date: 2 September 2008.

I am grateful to Jorma Merikoski for many helpful comments and remarks.

Construction	*Simplicity*	*Exactitude*	acc_0
DeTemple	15	10	0.002757
Ptolemy	16	10	0.003730
Labelle	15	10	0.002824
Hirano	19	13	0.003899
Richmond	45	26	0.004782
”Mascheroni”	34	21	0.009018

Table 1. Constructions of a regular pentagon

Various constructions aiming at the same goal have been compared by their simplicity values. However, the five operations concerned are geometrical and not arithmetic. Therefore the operations are incommensurable and the sum $m_1 + m_2 + n_1 + n_2 + n_3$ measures simplicity or complexity of the construction more or less arbitrarily.

Such a drawback is avoided in our approach where the complexity of a geometric construction is measured by its accuracy from a statistical point of view. I make a very natural assumption that each step in the construction is prone to measurement error. This error is specified by a simple statistical model.

The accuracy of a construction depends particularly on how precisely one can place the straightedge or the legs of the compass to a given point or to an intersection point created during the construction. In other respects, constructions are assumed to be accurate.

This approach corresponds more closely to Lemoine's *exactitude* equal to $m_1 + n_1 + n_2$ but it should be observed that in practice *simplicity* and *exactitude* are highly correlated as revealed by the following table describing properties of six different constructions of a regular pentagon. (These constructions are treated in detail later in Section 3.)

The correlation coefficient between *simplicity* and *exactitude* is 0.9969 so that there seems to be no significant difference in their capabilities to tell properties of constructions. [1]

The last column in the table tells accuracies given by the statistical model 0 (see below) mostly indicating rather good correspondence with Lemoine's measures in this case, but giving more precise information and showing, for example, that DeTemple's construction is better than Labelle's, although they both are equally simple in terms of geometrography. In the ”Mascheroni” construction made by using the compass only, the statistical measure indicates greater complexity than the geometrographic characteristics do.

Thinking statistically, it is natural to assume that the distribution of a placement (of either the straightedge or the compass) has a two-dimensional normal distribution around the correct point.

In theory, a crucial question is in case of a point of intersection of two lines, how the parameters of the distribution depend on the angle between the lines. The simplest choice is that the angle has no effect and the variance of the placement is

[1] This strong relation is revealed, for example, also by studying 89 constructions of Lemoine in [10] and giving correlation 0.9955.

a constant σ^2 in all directions in the plane (Model 0). However, in situations where the lines meet at a very small angle, there is a temptation to believe that typical variation depends on the angle and is greatest along the bisector and smallest in the orthogonal direction.

I have made experiments also with two alternative models (Models 1 and 2) which take the angle of intersection into account in a plausible way. These models are presented in Section 5. However, both of them lead to more or less severe contradictions and therefore it is safest to keep Model 0 as the main alternative.

We shall see that the choice of the basic error model is not critical since various models lead to very similar results, for example, when comparing accuracies of various constructions for the same target.

2. Some assumptions and basic operations

In this connection, inaccuracies are met only when locating points (of intersection) by the straightedge or the compass.

In order to fix scale and orientation in the plane, it is assumed that each construction starts from a fixed circle and one fixed point on it or from a fixed straight line and a circle with a fixed radius and with a center point on that line.

It is also assumed (as, for example, in [2]) that the compass is noncollapsing and so permits transferring of lengths.

Then the following basic constructions are available:

```
------------------------------------------------------------
P=point(x,y)           Set a point P with coordinates (x,y).
L=line(P,Q)            Draw a straight line L through points P and Q.
P=point_l(L,x,y)       Set a point (x,y) on line L.
E=edge(P,Q)            Measure the distance between points P and Q.
C=circle(P,R)          Draw a circle C with point P as the center
                       and R (constant or edge) as the radius.
C=circle_p(P,Q)        Draw a circle C with point P as the center
                       and going through point Q.
P=cross(L1,L2)         Set P as the point of intersection
                       between straight lines L1 and L2.
P=cross_cl(C,L,x,y)    Set P as the point of intersection
                       between circle C and straight line L
                       and being closest to a reference point (x,y).
P=cross_cc(C1,C2,x,y)  Set P as the point of intersection
                       between circles C1 and C2
                       and being closest to a reference point (x,y).
------------------------------------------------------------
```

The definitions are given in terms of commands for the GEOM program. I have made that program for creating geometric constructions and for computing accuracies of constructions according to alternative statistical models. GEOM also computes the geometrographic symbol, simplicity, and exactitude. GEOM works as a part of the Survo system [18]. In fact, GEOM creates separate files for points, lines, circles, and edges. The actual graphs are then plotted by the PLOT commands of Survo. [2]

By default, GEOM makes an 'exact' construction. By entering specifications

```
SAMPLES=N MODEL=<0,1, or 2>  STDDEV=s
```

[2] Usage of GEOM in Survo is described more thoroughly in Section 6.

GEOM repeats the construction N times using one of the models 0,1,2 for sampling the points with a value `s` for the basic standard deviation σ. Points given at the start of the construction are treated like points of intersection between two orthogonal lines. In this sampling mode, items given as fixed (without random variation) are introduced with a ’_’ in front of the label.

For example, the construction a perpendicular from a given point to a given straight line can be described as a series of commands in the following way.

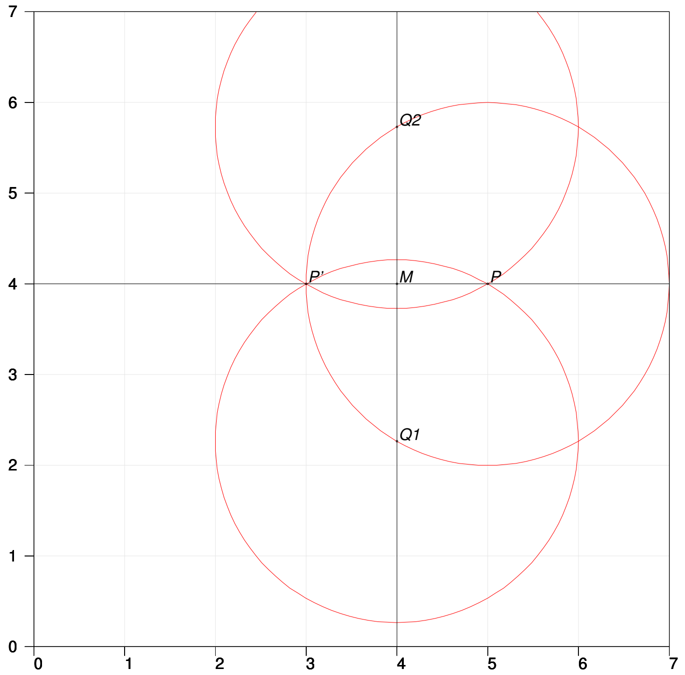


FIGURE 1. Perpendicular from a given point

The GEOM code given in the edit field of Survo:

```
____________________________________________________________
101 *  r1=2    r2=r1
102 *GEOM CUR+1,E,E+1
103 *Perpendicular
104 *p1=point(4,4)
105 *p2=point(4,6)
106 *_L1=line(P1,P2)
107 *P=point(5,4)
108 *symbol start
109 *C1=circle(P,*r1)
110 *Q1=cross_cl(C1,L1,4,2)
111 *Q2=cross_cl(C1,L1,4,6)
112 *C2=circle(Q1,*r2)
113 *C3=circle(Q2,*r2)
114 *P’=cross_cc(C2,C3,3,4)
```

```
115 *L=line(P,P')
116 *M=cross(L,L1)
117 *save point(M)
118 *save line(L)
119 E
120 *GEOM construction: Perpendicular
121 *Geometrographic symbol = 2S1+1S2+3C1+0C2+3C3
122 *Simplicity = 9  Exactitude = 5
------------------------------------------------------------
```

The GEOM command on line 102 activates the steps of the construction on lines from CUR+1 (103) to E (119). The first line tells the name of the construction 'Perpendicular'. Two points p1(4,4) and p2(4,6) are given on lines 104,105. A fixed line _L1 going through those points is defined on line 106. Objects given with lower case initials are not made visible when the graph generated by GEOM is drawn. Otherwise all notations are non-case-sensitive. Therefore, for example, p1 can later be referred to by P1.

Code 'symbol start' on line 108 tells that counting of geometrographic attributes will be (re)started from the next stage of the construction. Now the perpendicular from point P(5,4) (given on line 107) to line L1 is created by the following steps:

At first a circle C1 with center P and a 'suitable' radius r1 is drawn (line 109). An asterisk (*) in front of r1 merely informs that there is no need to measure the length of the radius and this fact is accounted for in geometrographic calculations only. The 'suitable' value of r1 is given as an extra specification (typical in Survo applications) on line 101.

Circle C1 intersects line L1 at two points close to points $(4, 2)$ and $(4, 6)$, respectively. These intersection points Q1 and Q2 are found by commands on lines 110 and 111.

Two circles C2, C3 with centers Q1, Q2 and a common 'suitable' radius r2 are drawn by commands on lines 112 and 113. In this case, r2 is set equal to r1 by the second specification on line 101.

The sought perpendicular is then obtained by drawing a straight line through points P and P' where the latter point is the point of intersection between circles C2 and C3 and 'close' to point (3,4). The point P' is found by the command on line 114 and the perpendicular L is drawn by the command on line 115.

The three last commands (lines 116–118) have no role in the exact construction. They are used for measuring the statistical accuracy of the construction. Here it is natural to check the intersection point M of lines L1 and L and the direction of the perpendicular L. Therefore coordinates of point M and points defining line L in each replication are saved in a text file (in Geom.TXT by default).

The last (optional) parameter (here E+1) of the GEOM command tells where to write a summary of geometrografic measures. These results of GEOM appear now on lines 120–122.

When the same code is activated by GEOM command from the following setup

```
____________________________________________________________
100 * SAMPLES=100000 MODEL=0 STDDEV=0.001 RAND=2008
101 *  r1=2    r2=r1
102 *GEOM CUR+1,E,E+1
103 *Perpendicular 1
104 *p1=point(4,4)
105 *p2=point(4,6)
...
117 *save point(M)
118 *save line(L)
119 E
____________________________________________________________
```

then (according to line 100) the construction is repeated 100'000 times and pertinent information obtained by save commands is saved in a text file Geom.TXT. In each replicate, when drawing straight lines and circles, essential points locating them are replaced by their sampled values according to the error model 0 and using the standard deviation $\sigma = 0.001$. The random deviates are created by the default random number generator of Survo with the seed value 2008.

The results saved in Geom.TXT are converted to a Survo data file K.SVO by FILE SAVE Geom.TXT TO NEW K and the first and last ten cases are

```
____________________________________________________________
                  MX        MY       LX1       LY1       LX2       LY2
     1     4.0000000 4.0006895 5.0001046 4.0009035 3.0000781 4.0004756
     2     4.0000000 3.9985637 5.0013286 4.0002385 3.0036884 3.9968974
     3     4.0000000 4.0003738 5.0009181 4.0014319 3.0012909 3.9993180
     4     4.0000000 3.9982571 5.0010213 3.9991808 2.9994477 3.9973339
     5     4.0000000 3.9995362 4.9991530 4.0005181 2.9991191 3.9985527
     6     4.0000000 3.9995679 4.9988252 3.9991844 2.9959532 3.9999535
     7     4.0000000 3.9991750 5.0005201 3.9998727 2.9999161 3.9984777
     8     4.0000000 4.0008971 4.9979912 4.0017919 2.9967416 3.9999976
     9     4.0000000 4.0000918 4.9971521 4.0010106 2.9987103 3.9991692
    10     4.0000000 4.0001340 4.9987566 4.0006284 2.9973038 3.9996376
   ...     ...       ...       ...       ...       ...       ...
 99991     4.0000000 4.0003325 4.9992809 3.9996596 2.9977360 4.0010074
 99992     4.0000000 4.0000661 5.0024009 4.0007761 2.9984049 3.9993565
 99993     4.0000000 4.0005191 5.0002210 4.0017158 2.9985739 3.9993210
 99994     4.0000000 3.9984695 5.0017009 3.9990228 3.0011519 3.9979178
 99995     4.0000000 4.0001657 5.0001185 3.9999611 3.0014406 4.0003700
 99996     4.0000000 4.0016463 4.9997410 4.0013076 2.9991154 4.0019855
 99997     4.0000000 4.0008671 5.0002880 4.0011284 3.0046015 4.0006072
 99998     4.0000000 3.9986675 4.9980187 3.9998269 2.9999949 3.9975058
 99999     4.0000000 3.9992650 4.9989538 4.0000202 3.0004692 3.9985094
100000     4.0000000 4.0001196 4.9976813 4.0004721 2.9991916 3.9997660
____________________________________________________________
```

MX and MY are coordinates of the point of intersection M. The remaining columns describe the sampled perpendicular L by coordinates of two points locating on that perpendicular.

The accuracy of this construction may be studied in Survo by making the following summary.

```
------------------------------------------------------------
*
*FILE SAVE GEOM.TXT TO NEW K
*
*DY=MY-4                      / vertical deviation of midpoint
*A=arctan((LY2-LY1)/(LX2-LX1)) / direction angle of perpendicular
*VAR DY:8,A:8 TO K
*
*MINSTAT K,CUR+1 / VARS=DY,A    RESULTS=0
*Basic statistics of data K N=100000
*Variable   mean      stddev          N  minimum   maximum
* DY       -0.000003  0.000957   100000 -0.004125  0.003979
* A         0.000001  0.000958   100000 -0.003934  0.003926
*
------------------------------------------------------------
```

The estimates of the midpoint and the direction of the perpendicular seem to be unbiased and the standard deviation of `DY` is close to the nominal error 0.001.

By letting `r1` grow and setting `r2=`k`*r1` where $k > 1$ the standard deviation of `DY` tends to 0.001 (the nominal error) and that of the direction angle tends to zero which is indicated by the following table where the values are computed from samples of size 100'000 and for $k = 2$.

r1	r2	sd(DY)	sd(angle)
2	2*r1	0.000909323	0.0005177181
3	2*r1	0.000900862	0.0003648293
4	2*r1	0.000909574	0.0002837157
5	2*r1	0.000919430	0.0002324257
6	2*r1	0.000928066	0.0001969223
10	2*r1	0.000955874	0.0001223272
100	2*r1	0.000993541	0.0000128590
1000	2*r1	0.000999348	0.0000012926
10000	2*r1	0.000999199	0.0000001293
100000	2*r1	0.000999251	0.0000000129
1000000	2*r1	0.000999257	0.0000000013

The accuracy of the Y coordinate of point `M` (`sd(DY)`) is slightly below the nominal error 0.001 for moderate lengths of radius `r1` but on the other hand the error in the direction of the perpendicular (`sd(angle)`) is significant for those values. For example, the best value for construction of point `M` seems to be about `r1=3` but for a point (2,4) locating 2 units to the left from point `M` the same construction would give 0.00111 as a standard deviation of the Y coordinate. This is as much above the nominal error as that of `M` was below the nominal error. Thus there is no uniformly best way to gain maximum accuracy for the perpendicular.

Since it is not plausible to set limits for the size of the area for constructions and the nominal level of accuracy can be achieved for *any point* on the perpendicular by using 'sufficiently' large circles in the construction, the following convention is adopted.

A perpendicular from a point P to a line L can be drawn without any error in the direction angle and its distance from the true perpendicular is specified by the error distribution of P.

This convention corresponds to a practice of using *a square ruler* in construction of a perpendicular from point P to line L. Then it is assumed that the first edge of the square ruler can be positioned strictly parallel to L and the second (orthogonal) edge through P like a standard ruler. [3]

Similarly, it is possible to draw a parallel to a line L from a given point P without any error in the direction angle and with an accuracy specified by the error distribution of point P in the distance between line L and its parallel.

Furthermore, by using large circles, it is possible to bisect an angle of two lines so that the direction angle of the bisector is correct but its distance from the true bisector has the error specified by the point of intersection.

Similarly, it is assumed that the midpoint between two given points can be found strictly on the straight line between those points and along that line with the accuracy specified by error distributions of the end points.

These observations permit simplifying more complicated constructions by introducing some composite operations in the `GEOM` program as follows.

```
_____________________________________________________________
L'=perpendicular(L,P)  Draw from point P perpendicular to line L.
L'=parallel(L,P)       Draw from point P parallel to line L.
L=bisect_a(L1,L2,a)    Draw bisector of angle between lines L1,L2
                       with (approximate) direction angle a in radians.
M=midpoint(P,Q,L)      Determine the midpoint between points P and Q
                       on line L.
_____________________________________________________________
```

For example, the previous construction of a perpendicular can be compressed into the form

```
_____________________________________________________________
101 * SAMPLES=100000 MODEL=0 STDDEV=0.001 RAND=2008
102 *GEOM CUR+1,E
103 *Perpendicular 1
104 *p1=point(4,4)
105 *p2=point(4,6)
106 *_L1=line(P1,P2)
107 *P=point(5,4)
108 *L=perpendicular(L1,P)
109 *M=cross(L,L1)
110 *save point(M)
111 E
112 *Summary of the above construction:
113 *FILE SAVE Geom.TXT TO NEW K
114 *MINSTAT K,CUR+1 / VARS=MX,MY
115 *Basic statistics of data K N=100000
116 *Variable   mean      stddev          N  minimum   maximum
117 * MX        4.000000  0.000000   100000  4.000000  4.000000
118 * MY        3.999992  0.000999   100000  3.995265  4.004154
_____________________________________________________________
```

[3] Interestingly, Lemoine [11] used square (ruler) as an extra tool in some of his constructions.

As another example, consider the construction of the midpoint of a given line segment. Without the midpoint command, it can be presented as follows:

```
____________________________________________________________
* SAMPLES=100000 MODEL=0 STDDEV=0.001 RAND=2008
* r=2
*GEOM CUR+1,E
*Midpoint of PQ
*P=point(0,0)
*Q=point(0,2)
*_L1=line(P,Q)
*C1=circle(P,*r)
*C2=circle(Q,*r)
*P'=cross_cc(C1,C2,1,1)
*Q'=cross_cc(C1,C2,-1,1)
*L=line(P',Q')
*M=cross(L,L1)
*save point(M)
E
___________________________________________________________
```

The summary of statistical variation of coordinates MX and MY of the midpoint

```
____________________________________________________________
*FILE SAVE Geom.TXT TO NEW K
*MINSTAT K,CUR+1
*Basic statistics of data K N=100000
*Variable   mean      stddev          N  minimum   maximum
* MX        0.000000  0.000000   100000  0.000000  0.000000
* MY        0.999999  0.000999   100000  0.995231  1.004327
___________________________________________________________
```

indicates that MX is accurate and the estimate of MY is unbiased with nominal standard deviation. In fact, when using Model 0 the same accuracy is obtained for all possible values r > 1.

By using the midpoint command the same task is carried out as follows

```
____________________________________________________________
* SAMPLES=100000 MODEL=0 STDDEV=0.001 RAND=2008
*GEOM CUR+1,E,G
*Midpoint of PQ
*P=point(0,0)
*Q=point(0,2)
*_L1=line(P,Q)
*M=midpoint(P,Q,L1)
*save point(M)
E
*FILE SAVE Geom.TXT TO NEW K
*MINSTAT K,CUR+1
*Basic statistics of data K N=100000
*Variable   mean      stddev          N  minimum   maximum
* MX        0.000000  0.000000   100000  0.000000  0.000000
* MY        0.999999  0.000999   100000  0.995231  1.004327
*
___________________________________________________________
```

and it leads to the same results.

3. Constructions of regular pentagons

Assume that a fixed circle is given and the task is to construct an inscribed regular pentagon. Six at least partially different constructions are now presented.

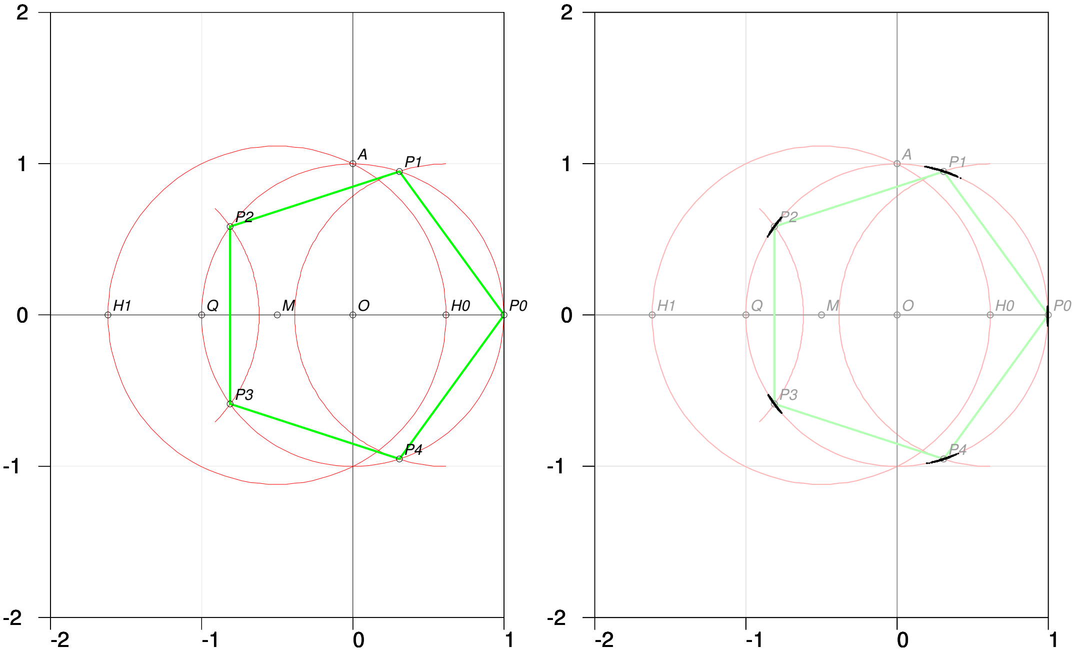


Figure 2. Construction of DeTemple, Sample of 1000 vertices on the right

3.1. **DeTemple** [2].

This construction is based on Carlyle circles and it is described by the following GEOM code.

```
____________________________________________________________
*GEOM CUR+1,E,E+1
*Pentagon (DeTemple)
*O=point(0,0)
*Q=point(-1,0)
*_C1=circle(O,1)
*LY0=line(O,_Q)
*P0=cross_cl(C1,LY0,1,0)
*LX0=perpendicular(LY0,*O)
*A=cross_cl(C1,LX0,0,1)
*
*symbol start
*M=midpoint(*O,Q,LY0)
*C5=circle_p(M,A)
*H0=cross_cl(C5,LY0,1,0)
```

```
115 *H1=cross_cl(C5,LY0,-1.5,0)
116 *C6=circle(H1,1,-0.25,0.25)
117 *P2=cross_cc(C1,C6,-1,0.5)
118 *P3=cross_cc(C1,C6,-1,-0.5)
119 *C7=circle(H0,*1,0.5,1.5)
120 *P1=cross_cc(C1,C7,0,1)
121 *P4=cross_cc(C1,C7,0,-1)
122 *
123 *E=edge(P0,P1)
124 *E1=edge(P1,P2)
125 *E2=edge(P2,P3)
126 *E3=edge(P3,P4)
127 *E4=edge(P4,P0)
128 *save edge(E)
129 *save point(P0)
130 *save point(P1)
131 *save point(P2)
132 *save point(P3)
133 *save point(P4)
134 E
135 *GEOM construction: Pentagon (DeTemple)
136 *Geometrographic symbol = 2S1+1S2+8C1+0C2+4C3
137 *Simplicity = 15  Exactitude = 10
------------------------------------------------------------
```

This construction has been repeated one million times by using the following specifications:

```
RAND=2008 SAMPLES=1000000 STDDEV=0.001   MODEL=0
```

Since essential information is saved for each replicate by `save` commands on lines 128–133, statistical variation can then be studied in Survo as follows:

```
-------------------------------------------------------------
141 *FILE SAVE Geom.TXT TO NEW K
142 *
143 *Constants related to accurate vertices (algebraic expressions):
144 *     f=(sqrt(5)+1)/2   "golden ratio"
145 *     r1x=1/2*(f-1)
146 *     r1y=1/2*sqrt(2+f)
147 *     r2x=-f/2
148 *     r2y=1/2*sqrt(3-f)
149       edge length: sqrt(3-f)=1.1755705045849
150 *
151 * Coordinates of vertices    Squared errors of vertices
152 * P0: p0X=1    p0Y=0         d0=(P0X-p0X)^2+(P0Y-p0Y)^2
153 * P1: p1X=r1x  p1Y=r1y       d1=(P1X-p1X)^2+(P1Y-p1Y)^2
154 * P2: p2X=r2x  p2Y=r2y       d2=(P2X-p2X)^2+(P2Y-p2Y)^2
155 * P3: p3X=r2x  p3Y=-r2y      d3=(P3X-p3X)^2+(P3Y-p3Y)^2
156 * P4: p4X=r1x  p4Y=-r1y      d4=(P4X-p4X)^2+(P4Y-p4Y)^2
157 *
158 *Computing total squared error:
159 *VAR D2:8=d0+d1+d2+d3+d4 TO K
160
161 *MINSTAT K,CUR+1  / VARS=E,D2
162 *Basic statistics of data K N=1000000
```

```
163 *Variable   mean         stddev           N  minimum      maximum
164 * E        1.175573756  0.002266360  1000000  1.165463791  1.186834877
165 * D2       0.000037995  0.000037299  1000000  0.000000036  0.000578814
166 *
167 *Accuracy acc0:
168 *sqrt(MAT_STAT.M(D2,mean)/5)=0.00275663698572
-------------------------------------------------------------
```

In these calculations, the accuracy of the construction is studied by two statistics `E` (length of the first edge of the pentagon) and `D2` (squared total error of the locations of the vertices). A suitable measure of the accuracy of the entire construction is the square root of the mean of `D2` divided by the number of vertices 5 and it is denoted by acc_0 (`acc0`).

The mean of `E` is very close to the expected value and, in general, it can be seen that the estimates of vertices are practically unbiased. The nominal error was `STDDEV=0.001` and then acc_0 is about 2.76 times this accuracy. In fact, acc_0 seems to be closely proportional to the nominal error as one can see from

`STDDEV`	acc_0
0.01	0.0275959310
0.001	0.0027566369
0.0001	0.0002756604
0.00001	0.0000275660
0.000001	0.0000027566

Table 2. Accuracy of the construction for various nominal errors

The right hand side of Figure 2 shows the distribution of vertices when 1000 replicates with `STDDEV=0.01` are generated and plotted. Points $P1$ and $P4$ have a larger variance than points $P2$ and $P3$.

3.2. **Ptolemy** [2], [14]**.** The classical construction presented by Ptolemy is created by the following GEOM code. This construction proceeds in the same way as DeTemple's to line 114.

```
-------------------------------------------------------------
101 *GEOM CUR+1,E,E+1
102 *Ptolemy
103 *O=point(0,0)
104 *Q=point(-1,0)
105 *_C1=circle(O,1)
106 *LY0=line(O,_Q)
107 *P0=cross_cl(C1,LY0,1,0)
108 *LX0=perpendicular(LY0,*O)
109 *A0=cross_cl(C1,LX0,0,1)
110 *
111 *symbol start
112 *M=midpoint(*O,Q,LY0)
113 *C5=circle_p(M,A0)
114 *H0=cross_cl(C5,LY0,1,0)
115 *C6=circle_p(A0,H0,1.15,1.85)
```

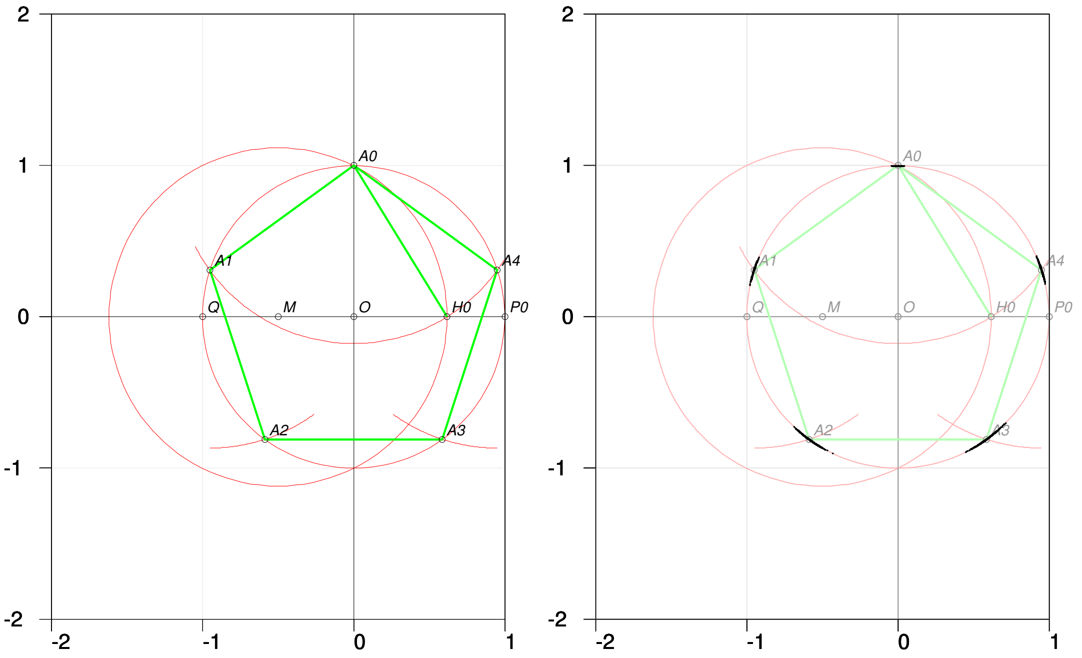


FIGURE 3. Construction of Ptolemy. Sample of 1000 vertices on the right

```
*A1=cross_cc(C1,C6,-1,0.5)
*A4=cross_cc(C1,C6,1,0.5)
*E=edge(A0,H0)
*C7=circle(A1,*E,1.5,1.7)
*A2=cross_cc(C1,C7,-0.5,-1)
*C8=circle(A4,*E,1.3,1.5)
*A3=cross_cc(C1,C8,0.5,-1)
*
*E0=edge(A0,A1)
*E1=edge(A1,A2)
*E2=edge(A2,A3)
*E3=edge(A3,A4)
*E4=edge(A4,A0)
*save edge(E)
*save point(A0)
*save point(A1)
*save point(A2)
*save point(A3)
*save point(A4)
E
*GEOM construction: Ptolemy
*Geometrographic symbol = 2S1+1S2+8C1+0C2+5C3
```

```
*Simplicity = 16  Exactitude = 10
____________________________________________________________
```

This construction, with the same parameters as in DeTemple's case, leads to a summary

```
_____________________________________________________________
*
*MINSTAT K,CUR+1  / VARS=E,D2
*Basic statistics of data K N=1000000
*Variable   mean         stddev             N  minimum      maximum
* E         1.175573796  0.002122005  1000000  1.164694915  1.185270556
* D2        0.000069550  0.000077151  1000000  0.000000027  0.001345038
*
*Accuracy acc0:
*sqrt(MAT_STAT.M(D2,mean)/5)=0.0037296054146
____________________________________________________________
```

Although the standard deviation of the edge length `E` is slightly smaller (0.002122 vs. 0.002266) the overall accuracy acc_0 (0.00373 vs. 0.00276) is definitely worse. Especially vertices $A2$ and $A3$ are more inaccurate since they have to be constructed through vertices $A1$ and $A4$ (see the right hand side of Fig. 3).

3.3. **Labelle** [9]**.** This construction can be realized by the following GEOM code.

```
_____________________________________________________________
*GEOM CUR+1,E,E+1
*Labelle
*A=point(0,0)
*P0=point(1,0)
*_C1=circle_p(A,P0)
*L1=line(A,_P0)
*B=cross_cl(C1,L1,-1,0)
*symbol start
*C2=circle_p(P0,B)
*C3=circle_p(B,A)
*C=cross_cc(C3,C2,-1,1)
*C4=circle_p(C,P0)
*D=cross_cl(C2,L1,3,0)
*E=cross_cc(C4,C2,1,2)
*C5=circle_p(D,E)
*F=cross_cc(C2,C4,-1,-1)
*C6=circle_p(D,F)
*P1=cross_cc(C1,C5,0,1)
*P4=cross_cc(C1,C5,0,-1)
*P2=cross_cc(C1,C6,-1,0.5)
*P3=cross_cc(C1,C6,-1,-0.5)
*E=edge(P0,P1)
*save edge(E)
*E1=edge(P1,P2)
*E2=edge(P2,P3)
*E3=edge(P3,P4)
*E4=edge(P4,P0)
*save point(P0)
*save point(P1)
*save point(P2)
```

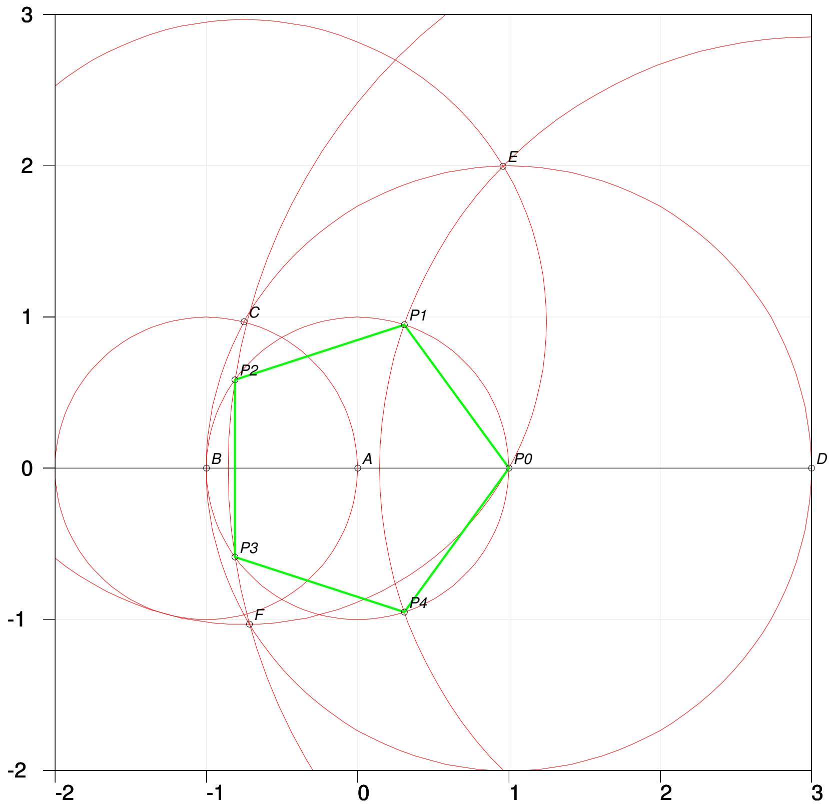


FIGURE 4. Construction of Labelle

```
*save point(P3)
*save point(P4)
E
*Geometrographic symbol = 0S1+0S2+10C1+0C2+5C3
*Simplicity = 15  Exactitude = 10
------------------------------------------------------------
```

Geometrographically, this solution is as good as DeTemple's. Repeating this construction with the same parameters as the previous ones leads to a summary

```
-------------------------------------------------------------
*
*MINSTAT K,CUR+1  / VARS=E,D2
*Basic statistics of data K N=1000000
*Variable   mean         stddev            N  minimum      maximum
* E         1.175570938  0.002450609   100000  1.164790116  1.185528960
* D2        0.000040084  0.000040707   100000  0.000000029  0.000542982
*
*Accuracy acc0:
```

```
*sqrt(MAT_STAT.M(D2,mean)/5)=0.00283139252151
____________________________________________________________
```

The overall accuracy acc_0 according to Model 0 is almost as good as DeTemple's. However, Models 1 and 2 (see Table 12) give less satisfactory results.

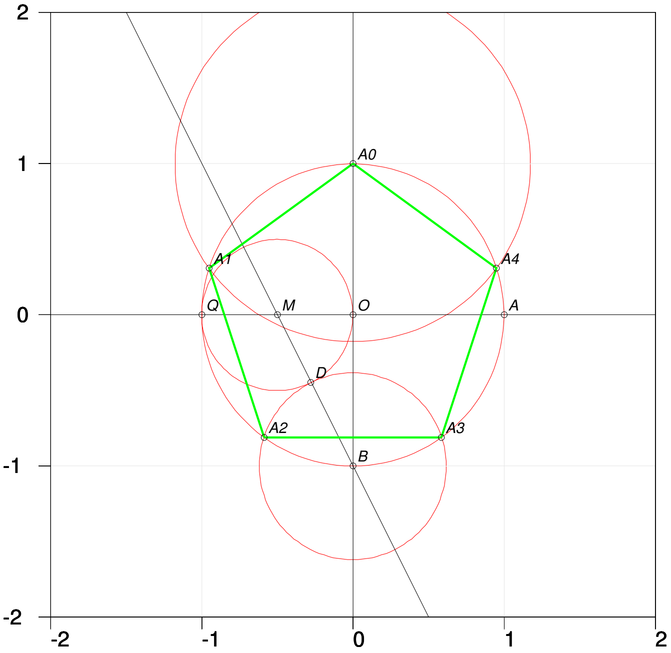


Figure 5. Construction of Hirano

3.4. **Hirano** [7]. This construction is created by the following GEOM code.

```
____________________________________________________________
*GEOM CUR+1,E,E+1
*Hirano
*O=point(0,0)
*Q=point(-1,0)
*_C1=circle(O,1)
*LY0=line(O,_Q)
*A=cross_cl(C1,LY0,1,0)
*LX0=perpendicular(LY0,*O)
*A0=cross_cl(C1,LX0,0,1)
*symbol start
*M=midpoint(*O,Q,LY0)
*B=cross_cl(C1,LX0,0,-1)
*LBM=line(B,M)
*C5=circle_p(M,O)
*D=cross_cl(C5,LBM,-0.2,-0.4)
```

```
*C6=circle_p(B,D)
*A2=cross_cc(C6,C1,-0.6,-0.8)
*A3=cross_cc(C6,C1,0.6,-0.8)
*E=edge(A2,A3)
*C7=circle(A0,E)
*A1=cross_cc(C7,C1,-0.8,0.6)
*A4=cross_cc(C7,C1,0.8,0.6)
*E1=edge(A0,A1)
*E2=edge(A1,A2)
*E3=edge(A3,A4)
*E4=edge(A4,A0)
*save edge(E)
*save point(A0)
*save point(A1)
*save point(A2)
*save point(A3)
*save point(A4)
E
*GEOM construction: Hirano
*Geometrographic symbol = 4S1+2S2+9C1+0C2+4C3
*Simplicity = 19  Exactitude = 13
------------------------------------------------------------
```

Repeating this construction with the same parameters as the previous ones leads to a summary

```
-------------------------------------------------------------
*
*MINSTAT K,CUR+1  / VARS=E,D2
*Basic statistics of data K N=1000000
*Variable   mean         stddev             N  minimum      maximum
* E        1.175565738  0.004044648  1000000  1.156263180  1.195719299
* D2       0.000075996  0.000088788  1000000  0.000000038  0.001442216
*
*Accuracy acc0:
*sqrt(MAT_STAT.M(D2,mean)/5)=0.00389860572035
------------------------------------------------------------
```

This construction is weaker than the previous ones according to all indicators.

3.5. **Richmond** [16]. The following GEOM code describes this construction.

```
-------------------------------------------------------------
*GEOM CUR+1,E,E+1
*Richmond
*O=point(0,0)
*B=point(-1,0)
*_C1=circle(O,1)
*LY0=line(O,_B)
*LX0=perpendicular(LY0,*O)
*A0=cross_cl(C1,LX0,0,1)
*
*symbol start
*M=midpoint(*O,B,LY0)
*LAM=line(A0,M)
*LMQ=bisect_a(LAM,LY0,0)
```

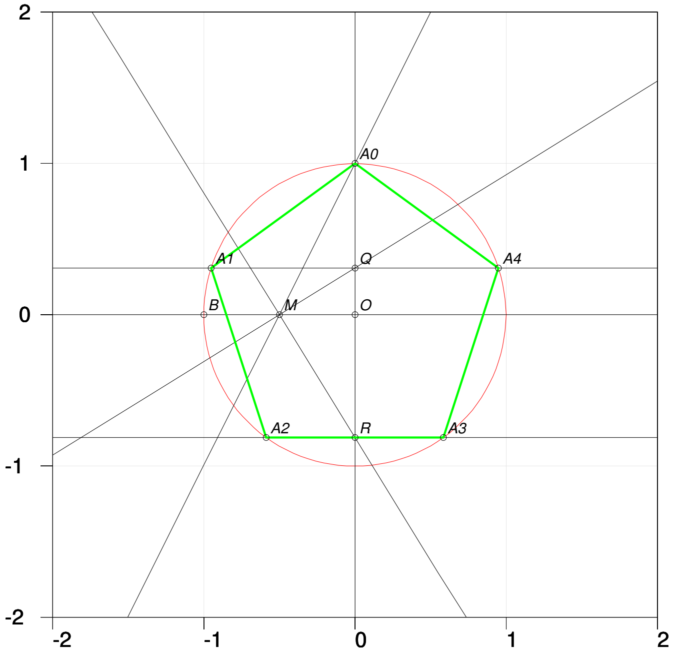


Figure 6. Construction of Richmond

```
*Q=cross(LMQ,LX0)
*LMR=perpendicular(LMQ,M)
*R=cross(LMR,LX0)
*LQ=perpendicular(LX0,Q)
*LR=perpendicular(LX0,R)
*A1=cross_cl(C1,LQ,-1,0.2)
*A4=cross_cl(C1,LQ,1,0.2)
*A2=cross_cl(C1,LR,-0.6,-0.8)
*A3=cross_cl(C1,LR,0.6,-0.8)
*
*E=edge(A0,A1)
*E1=edge(A1,A2)
*E2=edge(A2,A3)
*E3=edge(A3,A4)
*E4=edge(A4,A0)
*save edge(E)
*save point(A0)
*save point(A1)
*save point(A2)
*save point(A3)
*save point(A4)
E
*GEOM construction: Richmond
```

```
137 *Geometrographic symbol = 12S1+6S2+14C1+0C2+13C3
138 *Simplicity = 45  Exactitude = 26
-------------------------------------------------------------
```

Repeating this construction with the same parameters as the previous ones leads to a summary

```
--------------------------------------------------------------
159 *
160 *MINSTAT K,CUR+1  / VARS=E,D2
161 *Basic statistics of data K N=1000000
162 *Variable   mean         stddev             N  minimum      maximum
163 * E         1.175570922  0.002189156  1000000  1.165454618  1.186783847
164 * D2        0.000114350  0.000144151  1000000  0.000000013  0.002377883
165 *
166 *Accuracy acc0:
167 *sqrt(MAT_STAT.M(D2,mean)/5)=0.00478226135812
-------------------------------------------------------------
```

This construction is weaker than the previous ones according to all indicators.

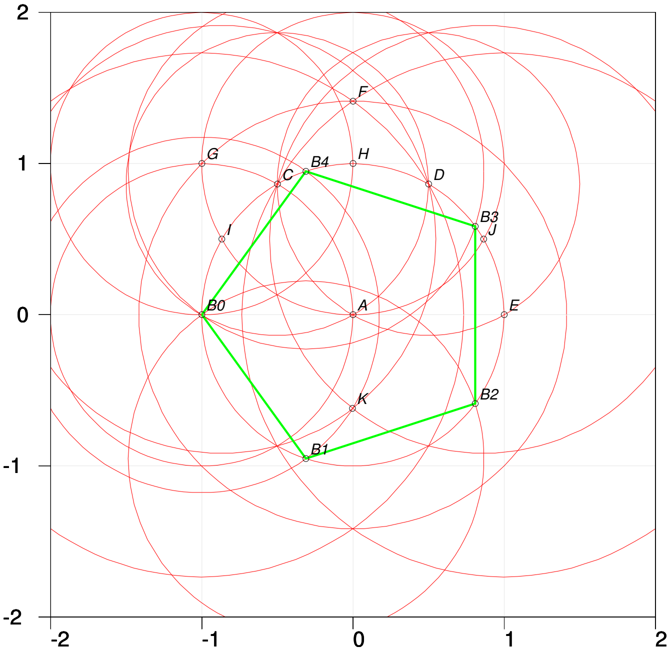


Figure 7. "Mascheroni" construction

3.6. **"Mascheroni"** [12]**.** This construction is made by using the compass as the only tool. The following GEOM code describes the solution.

```
--------------------------------------------------------------
101 *GEOM CUR+1,E,E+1
```

```
*"Mascheroni"
*A=point(0,0)
*B0=point(-1,0)
*_C1=circle(A,*1)
*
*symbol start
*C2=circle(B0,*1)
*C=cross_cc(C1,C2,-0.5,1)
*C3=circle(C,*1)
*D=cross_cc(C3,C1,0.5,1)
*C4=circle(D,*1)
*E=cross_cc(C4,C1,1,0)
*C5=circle_p(B0,D)
*b0d=edge(B0,D)
*C6=circle(E,*B0D)
*F=cross_cc(C5,C6,0,1.5)
*C7=circle_p(A,F)
*G=cross_cc(C7,C2,-1,1)
*C8=circle_p(G,B0)
*H=cross_cc(C8,C1,0,1)
*C9=circle_p(H,G)
*I=cross_cc(C9,C1,-1,0.5)
*J=cross_cc(C9,C1,1,0.5)
*C10=circle_p(I,D)
*C11=circle_p(J,*C)
*K=cross_cc(C10,C11,0,-0.7)
*C12=circle_p(B0,K)
*B4=cross_cc(C12,C1,-0.3,1)
*B1=cross_cc(C12,C1,-0.3,-1)
*C13=circle_p(B1,B0)
*C14=circle_p(B4,B0)
*B2=cross_cc(C13,C1,0.8,-0.6)
*B3=cross_cc(C14,C1,0.8,0.6)
*E=edge(B0,B1)
*E1=edge(B1,B2)
*E2=edge(B2,B3)
*E3=edge(B3,B4)
*E4=edge(B4,B0)
*save edge(E)
*save point(B0)
*save point(B1)
*save point(B2)
*save point(B3)
*save point(B4)
E
*GEOM construction: "Mascheroni"
*Geometrographic symbol = 0S1+0S2+21C1+0C2+13C3
*Simplicity = 34  Exactitude = 21
------------------------------------------------------------
```

Repeating this construction with the same parameters as the previous ones leads to a summary

```
-------------------------------------------------------------
```

```
*
*MINSTAT K,CUR+1  / VARS=E,D2
*Basic statistics of data K N=1000000
*Variable   mean         stddev             N  minimum      maximum
* E         1.175577809  0.005199012  1000000  1.146569437  1.201798513
* D2        0.000406598  0.000551322  1000000  0.000000021  0.011215175
*
*Accuracy acc0:
*sqrt(MAT_STAT.M(D2,mean)/5)=0.00901773402213
------------------------------------------------------------
```

This construction is definitely more inaccurate than all the others according to statistical indicators. However, geometrographic measures find it better than Richmond's. Those measures do not register the propagating error in a long chain of construction steps typical in this "Mascheroni" solution.

4. Approximate circle squaring

For over 100 years it has been known that it is impossible to to construct a square with the same area as a given circle by using only a finite number of steps with compass and straightedge. This fact does not diminish the interest of creating approximate solutions. Of course, most of these constructions have been presented in earlier centuries, but also later, for example, Ramanujan [15] has given at least two proposals.

From a statistical point of view, approximate circle squaring is an interesting problem, since now we can see how various constructions work in presence of errors encountered during the construction process. When the nominal error is large, say $\sigma > 0.001$, high-precision approximations are not so good as expected since their accuracy is lost due to influence of random noise. The goodness of each construction depends essentially on how accurately one is able to work with the compass and the straightedge.

The error models presented in this compendium give a new possibility of comparing various historical and modern constructions on different levels of the nominal error.

In this context, the geometrographic characteristics have no significance and so they are omitted.

Some of the historical constructions are related to approximations (22/7 and 353/113) appearing as convergents of the π continued fraction. At first, three such constructions are presented. Osler [13] has shown how to construct geometrically approximations to π of arbitrary (theoretical) accuracy related to Vieta's famous infinite product. The remaining solutions considered here have no general theoretical background. Most of them just happen to be reasonable approximate constructions.

A good source for historical constructions is [17] appearing also in [1].

4.1. **Gelder** [17]**.** This construction is based on the observation that $355/113 = 3 + 4^2/(7^2 + 8^2)$. As in all subsequent constructions, the radius of the circle to be "squared" is 2. The following GEOM code makes the construction and produces the edge of the square displayed in green in Fig. 8.

```
____________________________________________________________
*GEOM CUR+1,E
*Gelder
*C=point(2,2)
*A=point(0,2)
*_C1=circle(C,2)
*LAC=line(_A,C)
*B=cross_cl(C1,LAC,4,2)
*LCD=perpendicular(LAC,C)
*D=cross_cl(C1,LCD,2,4)
*p1=midpoint(D,C,LCD)
*p2=midpoint(D,p1,LCD)
*E=midpoint(D,p2,LCD)
*LAE=line(A,E)
*p3=midpoint(A,C,LAC)
*c2=circle_p(A,P3)
*F=cross_cl(C2,LAE,1,3)
*LFG=perpendicular(LAC,F)
*G=cross(LFG,LAC)
```

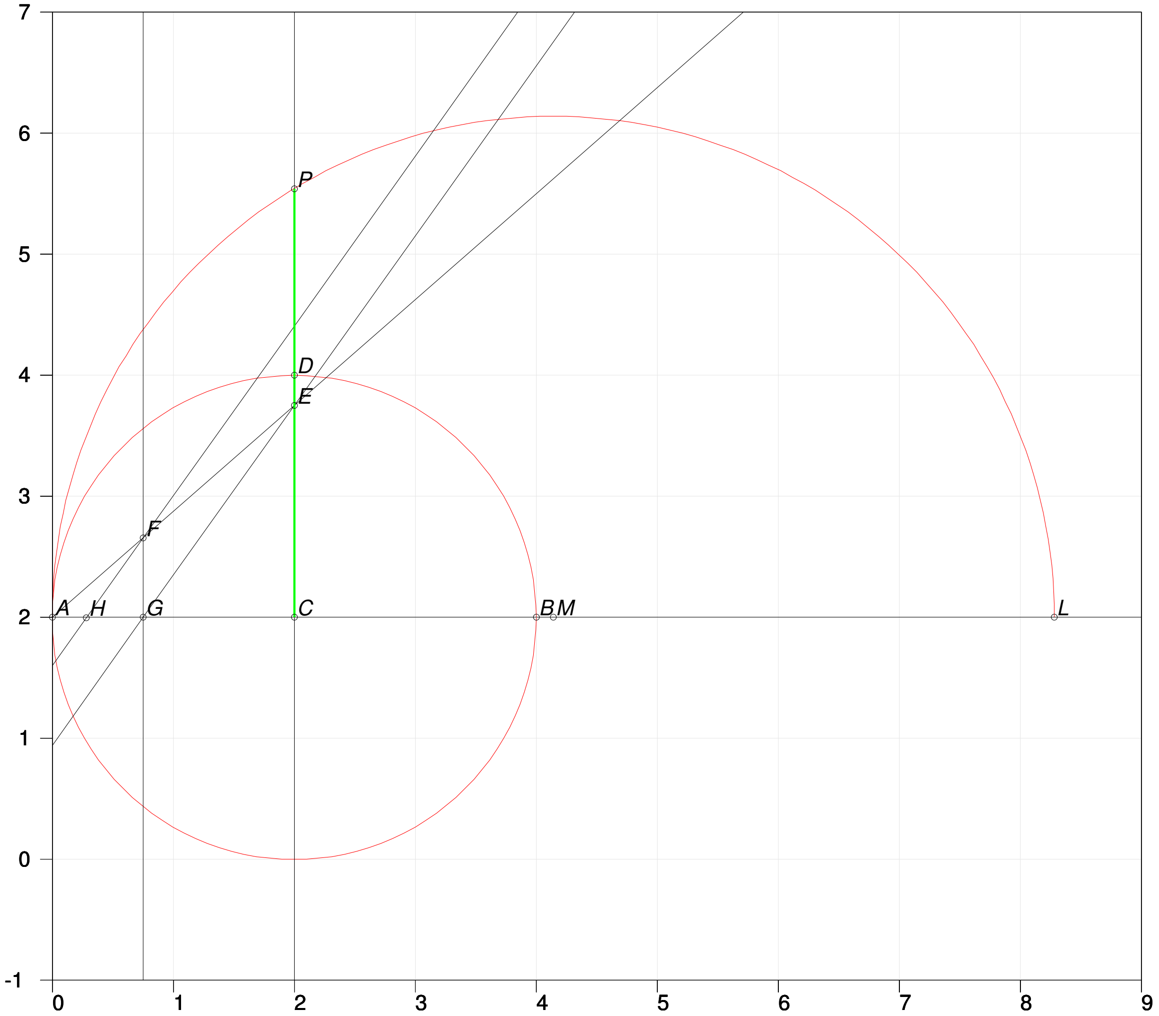


FIGURE 8. Construction of Gelder ($355/113 = 3.1415929\ldots$)

```
*LEG=line(E,G)
*LFH=parallel(LEG,F)
*H=cross(LFH,LAC)
*Ep=edge(A,H)
*c3=circle(B,*2)
*p4=cross_cl(C3,LAC,6,2)
*c4=circle(P4,*2)
*p5=cross_cl(C4,LAC,8,2)
*c5=circle(P5,Ep)
*L=cross_cl(C5,LAC,8.3,2)
*M=midpoint(A,L,LAC)
*C6=circle_p(M,A,0,1)
```

```
*P=cross_cl(C6,LCD,2,5.5)
*Edge=edge(C,P)
*save edge(Edge)
E
____________________________________________________________
```

This construction has been repeated 100'000 times by using the following specifications:

```
RAND=2008 SAMPLES=100000 STDDEV=0.002 MODEL=0
```

Since the radius of the circle is $r = 2$, σ =`STDDEV=0.002` corresponds to the nominal accuracy 0.001. The statistical accuracy is calculated as the RMSE (Root Mean Squared Error) of the ratio $X = (Edge - r\sqrt{\pi})/\sigma$ giving the proportional accuracy of the edge of the square as a multiple of σ. This is done by means of Survo as follows:

```
____________________________________________________________
*FILE SAVE Geom.TXT TO NEW GELDER
*VAR X:8=(Edge-2*sqrt(pi))/STDDEV TO GELDER / pi=3.141592653589793
*
*MINSTAT GELDER,CUR+1 / VARS=X
*Basic statistics of data GELDER N=100000
*Variable   mean         stddev             N  minimum      maximum
* X         0.005773709  2.672587812   100000 -10.87542871  12.45696825
*
*RMSE=sqrt(MAT_STAT.M(X,stddev)^2+MAT_STAT.M(X,mean)^2)
*
*RMSE=2.672959331
____________________________________________________________
```

Table 3 tells how the accuracy of Gelder's construction is related to nominal accuracy σ_1 = `STDDEV`$/r$.

σ_1	RMSE
10^{-2}	2.673
10^{-3}	2.673
10^{-4}	2.673
10^{-5}	2.673
10^{-6}	2.674
10^{-7}	2.776
10^{-8}	7.982

Table 3. Accuracy of the Gelder's construction for various nominal errors

The proportional accuracy of the construction stays constant (2.673) accuracy until the bias $355/113 - \pi = 0.00000026\ldots$ exceeds the nominal accuracy.

4.2. **Ramanujan 1** [15]. Ramanujan's first construction aims at the same approximation as Gelder's but it seems to be more complicated.

```
____________________________________________________________
*GEOM CUR+1,E
```

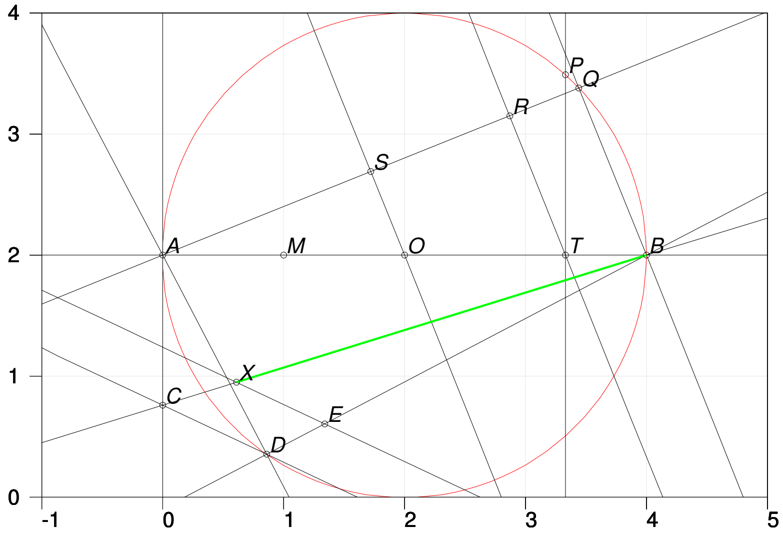


Figure 9. Construction of Ramanujan (355/113 = 3.1415929 . . .)

```
*Ramanujan 1
*O=point(2,2)
*A=point(0,2)
*_C1=circle(O,2)
*LAO=line(_A,O)
*B=cross_cl(C1,LAO,4,2)
*M=midpoint(A,O,LAO)
*c2=circle(B,2)
*p1=cross_cc(C1,C2,3,0)
*p2=cross_cc(C1,C2,3,4)
*c3=circle(P1,2)
*lP1O=line(P1,O)
*p3=cross_cl(C3,LP1O,4,-1.5)
*lP23=line(P2,P3)
*T=cross(LP23,LAO)
*LPT=perpendicular(LAO,T)
*P=cross_cl(C1,LPT,3,3)
*ept=edge(P,T)
*c4=circle(B,EPT)
*Q=cross_cc(C1,C4,3.5,3.5)
*LAQ=line(A,Q)
*LBQ=line(B,Q)
*LTR=parallel(LBQ,T)
*R=cross(LTR,LAQ)
*LOS=parallel(LBQ,O)
*S=cross(LOS,LAQ)
*c5=circle_p(A,S)
*D=cross_cc(C1,C5,0.5,0.5)
```

```
*LAD=line(A,D)
*LAC=perpendicular(LAO,A)
*ers=edge(R,S)
*c6=circle(A,ERS)
*C=cross_cl(C6,LAC,0,1)
*LBC=line(B,C)
*LBD=line(B,D)
*LCD=line(C,D)
*c7=circle_p(B,M)
*E=cross_cl(C7,LBD,1.5,0.5)
*LEX=parallel(LCD,E)
*X=cross(LEX,LBC)
*Edge=edge(B,X)
*save edge(Edge)
E
------------------------------------------------------------
```

By making 100'000 samples of this construction in the same way as in Gelder's case, the following summary for various nominal accuracies is obtained:

σ_1	RMSE
10^{-2}	3.970
10^{-3}	3.943
10^{-4}	3.943
10^{-5}	3.943
10^{-6}	3.943
10^{-7}	4.012
10^{-8}	8.488

Table 4. Accuracy of Ramanujan's 1^{st} construction for various nominal errors

These values are about 50 per cent higher than in Gelder's construction and thus Ramanujan's solution is less satisfactory.

4.3. **Solski.** This is an elegant historical construction aiming at the approximation 22/7. According to [20], it has been presented by Stanislaw Solski in 1683. The construction starts by dividing the radius into 7 equal parts, but the source does not reveal how Solski has done it. The most obvious solution is the following one:

```
------------------------------------------------------------
*GEOM CUR+1,E
*Solski
*/ d=0.55 dC=2+d dD=2+2*d dE=2+4*d dF=2+7*d
*O=point(2,2)
*A=point(4,2)
*_C1=circle(O,2)
*LAO=line(_A,O)
*LOB=perpendicular(LAO,O)
*B=cross_cl(C1,LOB,2,4)
*C=point(dC,2)
```

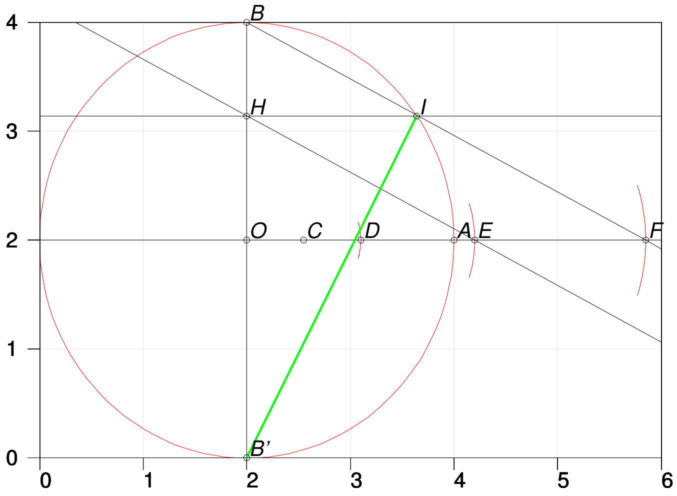


Figure 10. Construction of Solski ($22/7 = 3.142857\ldots$)

```
*C2=circle_p(C,O,-0.1,0.1)
*D=cross_cl(C2,LAO,dD,0)
*C3=circle_p(D,O,-0.1,0.1)
*E=cross_cl(C3,LAO,dE,0)
*C4=circle_p(E,C,-0.1,0.1)
*F=cross_cl(C4,LAO,dF,0)
*LBF=line(B,F)
*LEG=parallel(LBF,E)
*H=cross(LEG,LOB)
*LHI=perpendicular(LOB,H)
*I=cross_cl(C1,LHI,3.7,3)
*B'=cross_cl(C1,LOB,2,0)
*Edge=edge(B',I)
*save edge(Edge)
E
------------------------------------------------------------
```

The accuracy of the construction depends slightly on the parameter `d`. The best results are found by using very large values.

σ_1	RMSE
10^{-2}	1.893
10^{-3}	1.923
10^{-4}	4.027

Table 5. Accuracy of the Solski's construction for various nominal errors

Thus Solski's construction is more accurate than Gelder's for $\sigma_1 = 0.01$ and $\sigma_1 = 0.001$. In fact, when $\sigma_1 < 0.0002$, Gelder's construction will be more accurate.

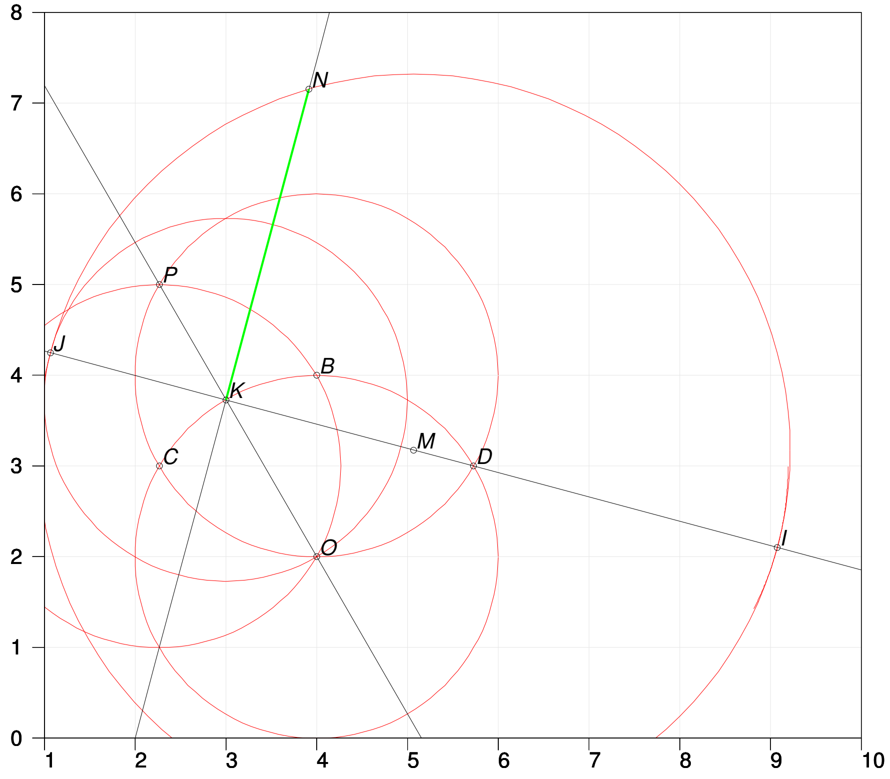


Figure 11. Construction according to Kühn

4.4. Construction of Kühn ($\sqrt{2}+\sqrt{3}=3.14626\ldots$). According to [11] this is a very old construction, attributed to Heinrich Kühn, in the *Novi Commentarii Acad. Petropol.*, Vol. III (1753).

The following GEOM code presents that construction.

```
____________________________________________________________
101 *GEOM CUR+1,E
102 *Kuhn
103 *O=point(4,2)
104 *B=point(4,4)
105 *_C1=circle(O,2)
106 *C2=circle(B,*2)
107 *C=cross_cc(C1,C2,2,3)
108 *D=cross_cc(C1,C2,6,3)
109 *C3=circle_p(C,B)
110 *P=cross_cc(C3,C2,2,5)
111 *LOP=line(O,P)
112 *K=cross_cl(C1,LOP,3,4)
113 *LDK=line(D,K)
114 *C4=circle_p(D,C,1.85,2)
115 *I=cross_cl(C4,LDK,9,2)
116 *C5=circle(K,2)
```

```
*J=cross_cl(C5,LDK,1,4)
*M=midpoint(I,J,LDK)
*C6=circle_p(M,I)
*LKN=perpendicular(LDK,K)
*N=cross_cl(C6,LKN,4,7)
*Edge=edge(K,N)
*save edge(Edge)
E
------------------------------------------------------------
```

However, already when $\sigma_1 = 0.01$ we get RMSE=3.757 indicating that this construction is definitely inferior to Solski's, for example.

In a web discussion
http://www.groupsrv.com/science/about13049.html
it has been proposed in 2004 that "sqrt(2)+sqrt(3) is the best one (=easiest) here. 5*10ˆ-3 is a *good* practical accuracy". The construction suggested there is different but statistically not more accurate than Kühn's.

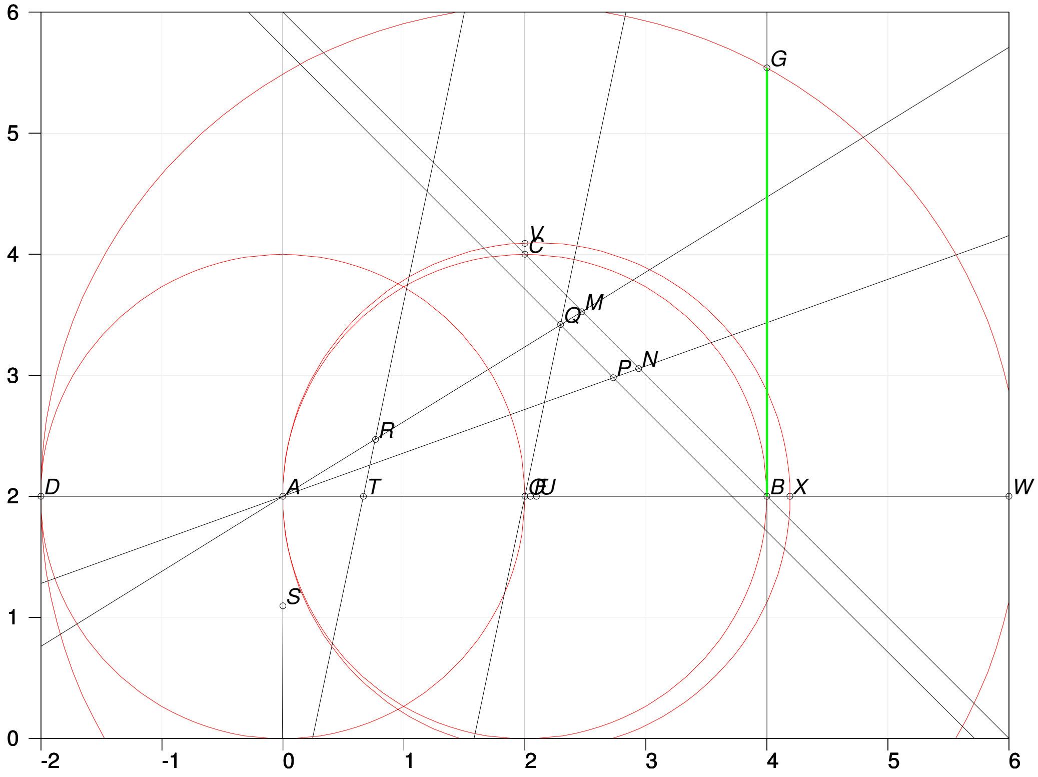


FIGURE 12. Construction of Ramanujan $((9^2 + 19^2/22)^{1/4} = 3.14159265258\ldots)$

4.5. **Ramanujan 2** [15]**.** Ramanujan's second construction is based on the observation that $\pi^4 = 97.409091034 \cdots \approx 97.40909090909 \cdots = 97 + 9/22 = 81 + 361/22$.

Nominally, it is more accurate than any of the previous ones, but it becomes acceptable only when σ_1 is very small since the construction is rather complicated.

```
____________________________________________________________
*GEOM CUR+1,E
*Ramanujan2
*O=point(2,2)
*A=point(0,2)
*_C1=circle(O,2)
*LAO=line(_A,O)
*B=cross_cl(C1,LAO,4,2)
*LX2=perpendicular(LAO,O)
*C=cross_cl(C1,LX2,2,4)
*C2=circle(A,2)
*p1=cross_cc(C1,C2,1,4)
*p2=cross_cc(C1,C2,1,0)
*c3=circle(P1,*2)
*lP1O=line(P1,O)
*p3=cross_cl(C3,LP1O,0,5.5)
*lT=line(P2,P3)
*T=cross(LT,LAO)
*LBC=line(B,C)
*eat=edge(A,T)
*c4=circle(C,EAT)
*M=cross_cl(C4,LBC,2.5,3.5)
*c5=circle(M,EAT)
*N=cross_cl(C5,LBC,3,4)
*LAM=line(A,M)
*LAN=line(A,N)
*eam=edge(A,M)
*c6=circle(A,EAM)
*P=cross_cl(C6,LAN,3,3)
*LPQ=parallel(LBC,P)
*Q=cross(LPQ,LAM)
*LOQ=line(O,Q)
*LTR=parallel(LOQ,T)
*R=cross(LTR,LAM)
*D=cross_cl(C2,LAO,-2,2)
*LX0=perpendicular(LAO,A)
*c7=circle_p(A,R)
*S=cross_cl(C7,LX0,0,1)
*c8=circle_p(O,S)
*X=cross_cl(C8,LAO,4,2)
*U=midpoint(A,X,LAO)
*C9=circle_p(U,A)
*V=cross_cl(C9,LX2,2,4)
*eov=edge(O,V)
*c10=circle(B,EOV)
*W=cross_cl(C10,LAO,6,2)
*F=midpoint(D,W,LAO)
*C11=circle_p(F,D)
*LBG=perpendicular(LAO,B)
*G=cross_cl(C11,LBG,4,5.5)
```

```
*Edge=edge(B,G)
*save edge(Edge)
E
-----------------------------------------------------------
```

σ_1	RMSE
10^{-2}	4.105
10^{-3}	4.095
10^{-4}	4.095
10^{-5}	4.095
10^{-6}	4.095
10^{-7}	4.095
10^{-8}	4.096
10^{-9}	4.106
10^{-10}	4.987
10^{-11}	28.710

TABLE 6. Accuracy of Ramanujan's 2^{nd} construction for various nominal errors

According to Table 6 this construction is statistically more inaccurate than, for example, Gelder's for $\sigma_1 > 10^{-7}$.

4.6. **Construction based on Vieta's product.** It is shown by T.J.Osler in [13] how partial products of the well-known Vieta's formula

$$\frac{2}{\pi} = \sqrt{\frac{1}{2}}\sqrt{\frac{1}{2}+\frac{1}{2}\sqrt{\frac{1}{2}}}\sqrt{\frac{1}{2}+\frac{1}{2}\sqrt{\frac{1}{2}+\frac{1}{2}\sqrt{\frac{1}{2}}}}\cdots$$

can be used for creating geometric constructions of π and thus, in theory, approximate circle squaring is possible to any degree of precision.

At first glance, this approach does not seem to be very promising as one can see from the following recursive Survo calculation

```
-----------------------------------------------------------
*SAVE VIETA  / Cumulative values of Vieta's product
*
*n'th term in Vieta's product:
 T(n):=if(n=1)then(sqrt(p))else(sqrt(p+p*T(n-1)))   p=1/2
*
*Product of first n terms:
 P(n):=if(n=1)then(T(1))else(P(n-1)*T(n))
*
*Approximation of pi by first n terms:
*pi(n):=2/P(n)           bias d(n):=pi-pi(n)  pi=3.141592653589793
*
*ACCURACY=16
*Results obtained simultaneuosly by activating just one of them:
*  pi(1).=2.8284271247461898           d(1).=0.3131655288436033
*  pi(2).=3.0614674589207183           d(2).=0.0801251946690749
```

```
 16 *  pi(3).=3.121445152258052           d(3).=0.0201475013317411
 17 *  pi(4).=3.1365484905459389          d(4).=0.0050441630438542
 18 *  pi(5).=3.1403311569547525          d(5).=0.0012614966350406
 19 *  pi(6).=3.1412772509327729          d(6).=0.0003154026570202
 20 *  pi(7).=3.1415138011443009          d(7).=0.0000788524454922
 21 *  pi(8).=3.1415729403670913          d(8).=0.0000197132227018
 22 *  pi(9).=3.1415877252771591          d(9).=0.000004928312634
 23 * pi(10).=3.1415914215111997         d(10).=0.0000012320785934
 24 * pi(11).=3.1415923455701176         d(11).=0.0000003080196755
 25 * pi(12).=3.141592576584872          d(12).=0.0000000770049211
 26 * pi(13).=3.1415926343385627         d(13).=0.0000000192512304
 27 * pi(14).=3.1415926487769852         d(14).=0.0000000048128079
 28 * pi(15).=3.1415926523865907         d(15).=0.0000000012032024
 29 * pi(16).=3.1415926532889928         d(16).=0.0000000003008003
 30 * pi(17).=3.1415926535145933         d(17).=0.0000000000751998
 31 * pi(18).=3.1415926535709935         d(18).=0.0000000000187996
 32 * pi(19).=3.1415926535850933         d(19).=0.0000000000046998
 33 * pi(20).=3.1415926535886185         d(20).=0.0000000000011746
 34 * pi(21).=3.1415926535895           d(21).=0.0000000000002931
 35 * pi(22).=3.1415926535897203         d(22).=0.0000000000000728
 36 * pi(23).=3.1415926535897754         d(23).=0.0000000000000178
 37 * pi(24).=3.1415926535897891         d(24).=0.000000000000004
 38 * pi(25).=3.1415926535897927         d(25).=0.0000000000000004
 35 * pi(26).=3.1415926535897936         d(26).=-0.0000000000000004
 36 *
------------------------------------------------------------
```

showing that the convergence is rather slow.

When the number n of terms in the product is 4, Osler's construction, shown in Fig. 13 is created by the GEOM code

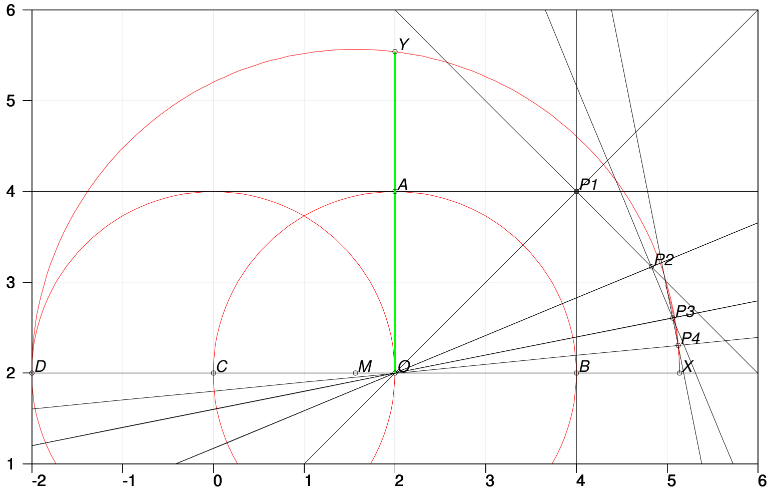


FIGURE 13. Osler-Vieta construction for $n = 4$

```
_____________________________________________________________
*GEOM CUR+1,E
*Osler - Vieta
*O=point(0,0)
*A=point(0,2)
*_C1=circle(O,2)
*LOA=line(O,_A)
*LOB=perpendicular(LOA,O)
*B=cross_cl(C1,LOB,2,0)
*LAP1=perpendicular(LOA,A)
*LBP1=perpendicular(LOB,B)
*P1=cross(LAP1,LBP1)
*
*L1=line(O,P1)
*LP1=perpendicular(L1,P1)
*LB2=bisect_a(L1,LOB,pi/8) / pi=3.141592653589793
*P2=cross(LB2,LP1)
*
*L2=line(O,P2)
*LP2=perpendicular(L2,P2)
*LB3=bisect_a(L2,LOB,pi/16)
*P3=cross(LB3,LP2)
*
*L3=line(O,P3)
*LP3=perpendicular(L3,P3)
*LB4=bisect_a(L3,LOB,pi/32)
*P4=cross(LB4,LP3)
*
*C2=circle_p(O,P4,0,0.1)
*X=cross_cl(C2,LOB,3,0)
*C=cross_cl(C1,LOB,-2,0)
*C3=circle_p(C,O)
*D=cross_cl(C3,LOB,-4,0)
*M=midpoint(D,X,LOB)
*C4=circle_p(M,D,0,1)
*Y=cross_cl(C4,LOA,0,3.5)
*Edge=edge(O,Y)  / sqrt(4*pi(4))
*save edge(Edge)
E
____________________________________________________________
```

In this construction, lines $OP_i,\ i = 1, 2, 3, 4$ have direction angles $\pi/2^{i+1}$. Point P_1 is $(2, 2)$ and points $P_i, i = 2, 3, 4$ are constructed so that line P_iP_{i-1} is perpendicular to line OP_{i-1}. Then the $OX = OP_4$ approximates π and the required edge of the square is obtained as the geometric mean of OD and OX.

The construction is easily extended for greater values of n. Table 7 shows the relative accuracies RMSE of Osler–Vieta construction for number of terms $n = 4, 6, 8, \ldots, 20$ in Vieta's product.

σ_1	4	6	8	10	12	14	16	18	20
10^{-2}	3.636	3.728	3.808	3.930	4.014	4.121	4.229	4.308	4.415
10^{-3}	7.338	3.747	3.806	3.930	4.013	4.120	4.229	4.307	4.415
10^{-4}		5.467	3.813	3.930	4.013	4.120	4.229	4.307	4.415
10^{-5}			4.545	3.934	4.013	4.120	4.229	4.307	4.415
10^{-6}				4.241	4.014	4.120	4.229	4.307	4.415
10^{-7}					4.130	4.120	4.229	4.307	4.415
10^{-8}						4.163	4.230	4.307	4.415
10^{-9}						7.373	4.248	4.307	4.415
10^{-10}							5.719	4.312	4.414
10^{-11}								4.922	4.414
10^{-12}									4.659

Table 7. Relative accuracies of Osler–Vieta construction for $n = 4, 6, 8, \ldots, 20$

For lower accuracies this construction is not good. Only when $\sigma_1 < 10^{-10}$ it becomes better than Ramanujan's second construction.

4.7. **Constructions based on results of random searches.** Current technology allows searching for approximations of π experimentally. I have made experiments by using a setup where the given circle of radius R locates in a square grid, for example, of size 4×4 as shown in Fig. 14.

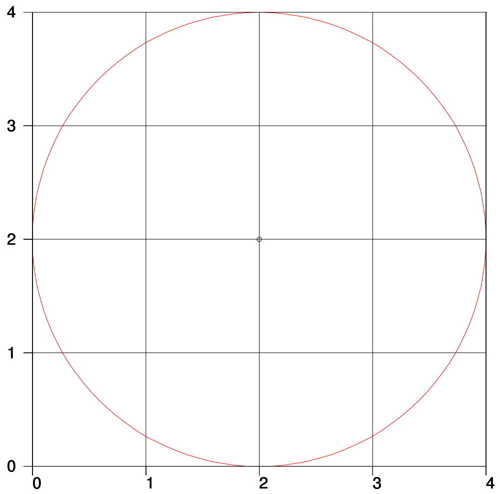


FIGURE 14. Circle of radius $R = 2$ in a 4×4 grid

Then straight lines through the grid points and circles with integer radius and center in a grid point are drawn and distances between points of intersection of these objects are computed. The target is to find situations where such a distance is as close as possible to $\sqrt{\pi}R$ and thus provides a good candidate for approximate circle squaring.

Here two such constructions called CL4 and CL42 are presented. The CL4 construction is based on an experiment where a line going through points (X_1, Y_1) and (X_2, Y_2) and a circle with radius r and center (X_0, Y_0) are drawn so that each of the coordinates is selected at random from numbers $0, 1, 2, 3, 4$ and the radius from numbers $1, 2, 3, 4$. The target is to find a point of intersection so that its distance from a fixed point $(2, 0)$ is as close to $2\sqrt{\pi} = 3.5449077018\ldots$ as possible.

By a simple program it is possible to find immediately that this procedure gives the best result, for example, by $X_1 = 4, Y_1 = 4, X_2 = 0, Y_2 = 3, X_0 = 3, Y_0 = 2, r = 3$ and the edge length of the square will then be $2\sqrt{(84 - 6\sqrt{26})/17} = 3.544868\ldots$ corresponding to a π approximation $(84 - 6\sqrt{26})/17 = 3.14152\ldots$.

This leads to a simple construction

```
____________________________________________________________
101 *GEOM CUR+1,E
102 *CL4
103 *O=point(2,2)
104 *A=point(2,0)
105 *_C1=circle(O,2)
```

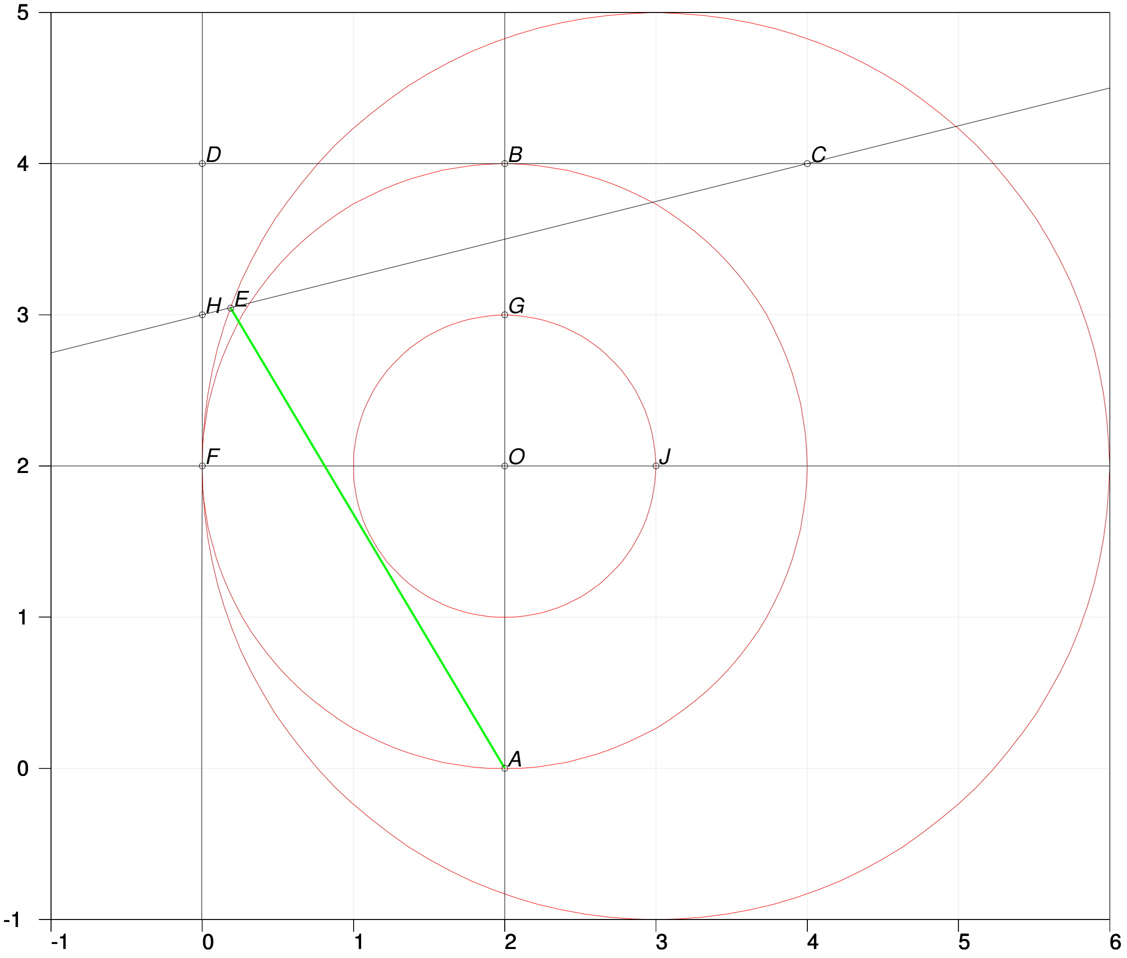


FIGURE 15. CL4 construction

```
106 *LX4=line(A,O)
107 *B=cross_cl(C1,LX4,2,4)
108 *c2=circle(B,*2)
109 *LY4=perpendicular(LX4,*B)
110 *C=cross_cl(C2,LY4,4,4)
111 *D=cross_cl(C2,LY4,0,4)
112 *LY2=perpendicular(LX4,O)
113 *LX2=perpendicular(LY4,*D)
114 *F=cross(LX2,LY2)
115 *G=midpoint(B,O,LX4)
116 *H=midpoint(D,F,LX2)
117 *C3=circle_p(O,G)
118 *J=cross_cl(C3,LY2,3,2)
```

```
*eag=edge(A,G)
*C4=circle(J,EAG)
*L=line(H,C)
*E=cross_cl(C4,L,0,3)
*Edge=edge(A,E)
*save edge(Edge)
E
-------------------------------------------------------------
```

and its accuracy is described by

σ_1	RMSE
10^{-2}	2.125
10^{-3}	2.125
10^{-4}	2.135
10^{-5}	2.906

Table 8. Accuracy of the CL4 construction for various nominal errors

In fact, the statistical accuracy of CL4 is better than Solski's for $\sigma_1 > 0.0003$ and better than Gelder's for $\sigma_1 < 0.0000125$.

Construction CL42 is based on the same setup but now two pairs consisting of a straight line and a circle are selected at random and the distances between intersection points is studied.

The best result can be obtained in several ways. The most suitable for actual construction seems to be the following

pair	X_1	Y_1	X_2	Y_2	X_0	Y_0	r
1	2	0	3	3	4	1	2
2	0	2	3	3	2	4	4

Table 9. Optimal pairs for the CL42 construction

The edge length of the approximating square will now be
$\sqrt{(380-17\sqrt{15})/25} = 3.54490780\ldots$
corresponding to the approximation
$\pi \approx (380-17\sqrt{15})/100 = 3.14159283\ldots.$
It is slightly better than $355/113 = 3.14159292\ldots$

If the CL42 construction is carried out according to the steps given by the GEOM code

```
-------------------------------------------------------------
*GEOM CUR+1,E
*CL42
*P40=point(4,0)
```

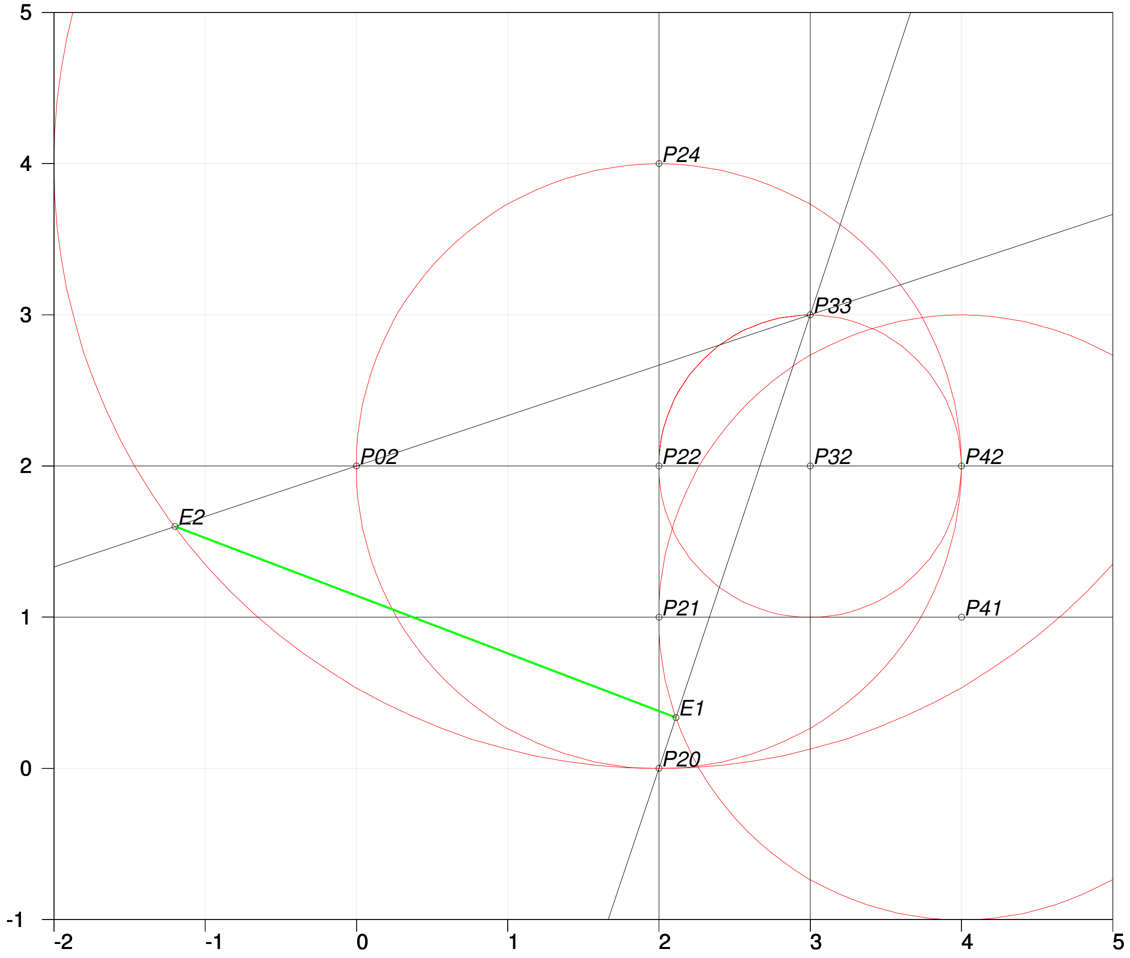


Figure 16. CL42 construction

```
*P42=point(4,2)
*_C=circle(P42,2)
*LX4=line(_P40,P42)
*P44=cross_cl(C,LX4,4,4)
*LY2=perpendicular(LX4,P42)
*P22=cross_cl(C,LY2,2,2)
*P62=cross_cl(C,LY2,6,2)
*P41=midpoint(P42,P40,LX4)
*LY1=perpendicular(LX4,P41)
*c2=circle(P41,*2)
*P61=cross_cl(C2,LY1,6,1)
*P52=midpoint(P42,P62,LY2)
*LX5=perpendicular(LY2,P52)
```

```
*C3=circle_p(P52,P42,0.5,1)
*P53=cross_cl(C3,LX5,5,3)
*L1=line(P40,P53)
*C4=circle(P61,*2)
*E1=cross_cl(C4,L1,4,0)
*L2=line(P22,P53)
*C5=circle_p(P44,P40)
*E2=cross_cl(C5,L2,1,2)
*Edge=edge(E1,E2)
*save edge(Edge)
E
------------------------------------------------------------
```

the following statistical accuracies are found

σ_1	RMSE
10^{-2}	2.637
10^{-3}	2.636
10^{-4}	2.636
10^{-5}	2.636
10^{-6}	2.637
10^{-7}	2.681
10^{-8}	5.651

Table 10. Accuracy of the CL42 construction for various nominal errors

and they are a little better than those of Gelder's construction.

4.8. **Summary.** Table 11 tells the statistical accuracies of best constructions on various error levels according to Model 0.

σ_1	Solski	CL4	Gelder	CL42	Ramanujan 2	Osler – Vieta
10^{-2}	1.893	2.125	2.673	2.637	4.105	3.636 (4)
10^{-3}	1.923	2.125	2.673	2.636	4.095	3.747 (6)
10^{-4}	4.027	2.135	2.673	2.636	4.095	3.813 (8)
10^{-5}		2.906	2.673	2.636	4.095	3.934 (10)
10^{-6}			2.674	2.637	4.095	4.014 (12)
10^{-7}			2.776	2.681	4.095	4.120 (14)
10^{-8}			7.982	5.651	4.096	4.230 (16)
10^{-9}					4.106	4.248 (16)
10^{-10}					4.987	4.312 (18)
10^{-11}						4.414 (20)

Table 11. Accuracy of the best constructions for various nominal errors (Integer in the Osler – Vieta column is the number of terms.)

Fig. 17 shows how the best construction depends on the nominal statistical error.

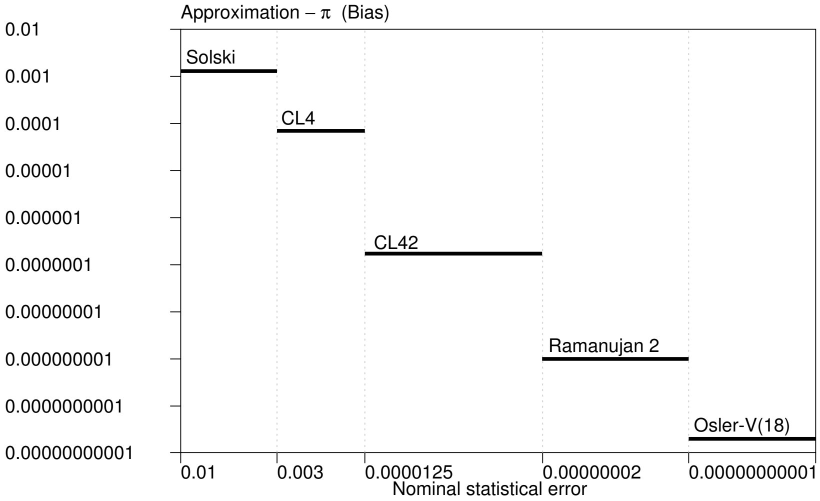


Figure 17. Best constructions in various intervals of the nominal error

5. Alternative definitions of the error distribution

5.1. **Model 1.** Let $P(x_0, y_0)$ be an intersection point of two lines with direction angles γ_1 and γ_2 $(-\pi/2 < \gamma_1 \le \gamma_2 \le \pi/2)$. Then the conditional error distribution of P is a two-dimensional normal distribution with the mean (x_0, y_0) and covariance matrix

$$\mathbf{\Sigma} = \mathbf{C}\mathbf{S}\mathbf{S}'\mathbf{C}'. \tag{1}$$

Here

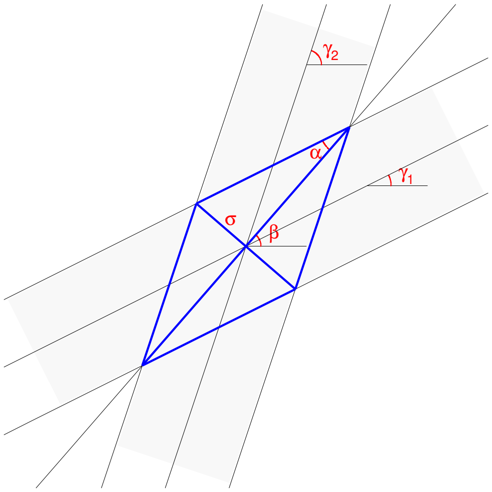
$$\mathbf{S} = \begin{bmatrix} \sigma/\tan\alpha & 0 \\ 0 & \sigma \end{bmatrix}, \tag{2}$$

$$\mathbf{C} = \begin{bmatrix} \cos\beta & -\sin\beta \\ \sin\beta & \cos\beta \end{bmatrix}. \tag{3}$$

and $\sigma > 0$ is a constant, β is the direction angle of the bisector of the acute angle between the lines, i.e.

$$\beta = \begin{cases} (\gamma_1 + \gamma_2)/2, & \text{if } \gamma_1 + \gamma_2 \le \pi/2, \\ (\gamma_1 + \gamma_2 - \pi)/2, & \text{if } \gamma_1 + \gamma_2 > \pi/2 \end{cases}$$

and α is one half of the acute angle between the lines. This setup becomes clear from the following figure.



The primary cause for inaccuracies is the fact that, in practice, lines are strips of a certain width. Then an ideal intersection point is characterized by a rhombus. The diagonals of such a rhombus are selected to be proportional to the standard deviations of the principal components of the error distribution.

Then the maximal standard deviation is $\sigma/\tan(\alpha)$ in direction of the first principal component (bisector of the acute angle) and the minimal one σ in direction of the second principal component.

Thus a sampled point (X, Y) representing the true point $P(x_0, y_0)$ in construction is

$$\begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} + \mathbf{CS} \begin{bmatrix} U_1 \\ U_2 \end{bmatrix} \tag{4}$$

where U_1 and U_2 are independent standard normal variables. The $\mathbf{S}$ matrix rescales variances and the $\mathbf{C}$ matrix rotates the point to direction β.

5.2. **Model 2.** The second alternative definition for the error distribution is the following one. Assuming that the distance D_1 of the sampled point (X, Y) from the first intersecting line has the $N(0, \sigma^2)$ distribution and its distance D_2 from the second line has the same distribution independently of the first one. Then (X, Y) is the intersection point of lines

$$\cos\gamma_i(Y - y_0) = \sin\gamma_i(X - x_0) + D_i, \quad i = 1, 2,$$

and thus we have

$$\begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} + \mathbf{A} \begin{bmatrix} D_1 \\ D_2 \end{bmatrix}$$

where

$$\mathbf{A} = \begin{bmatrix} \cos\gamma_2 & -\cos\gamma_1 \\ \sin\gamma_2 & -\sin\gamma_1 \end{bmatrix} / \sin(\gamma_2 - \gamma_1).$$

In this case, the covariance matrix $\mathbf{AA}'$ has the spectral decomposition

$$\mathbf{AA}' = \mathbf{CLC}'$$

where $\mathbf{L}$ is the diagonal matrix of eigenvalues

$$\mathbf{L} = \begin{bmatrix} \sigma^2/(2\sin^2\alpha) & 0 \\ 0 & \sigma^2/(2\cos^2\alpha) \end{bmatrix}.$$

and $\mathbf{C}$ is the same matrix of orthonormal eigenvectors as (3) in Model 1 (4). Since also the ratio of eigenvalues $1/\tan^2\alpha$ is the same as in (1), the form of the covariance structure is the same in both models.

However, the scale is different since in Model 1 (4) the standard deviation along the first principal axis is constant σ but in this latter Model 2 it is $\sigma/(\sqrt{2}\cos\alpha)$ and tends to $\sigma/\sqrt{2}$ as the angle 2α between the lines tends to 0.

Thus Model 2 seems to behave poorly due to the fact that $\mathbf{A}$ is almost singular if $\gamma_1 \approx \gamma_2$ and gives too low values for the standard deviation when the angle of intersection is small.

5.3. **Model 0.** The simplest alternative is the model

$$\begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} + \sigma \begin{bmatrix} U_1 \\ U_2 \end{bmatrix}$$

giving the same normal error distribution in each direction irrespective of the angle of intersection.

5.4. **Problems with very small angles.** Hudson [8], (pp. 109–111) discussed the problem of ill-defined intersections. She suggested 'to economize the chance of error' in those cases by extending the construction so that same points can be created as intersections 'at a moderate angle'. In fact, Hudson's treatise is the only source I have found where problems related to accuracy of geometric constructions are taken into consideration.

When studying Hudson's extended constructions, it turns out that especially in case of Model 1 it is possible to reduce the statistical error even when the lines meet at an angle of 30 degrees. Thus the true accuracy seems to be better than Model 1 predicts and tends to be closer to the accuracy given by Model 0.

So there are problems with Models 1 and 2 especially with very small angles. Since in Model 0 such difficulties are not met, it has been used as the main alternative in this paper. Limitation to this simplest model is also supported by the experience that all these three models lead to similar results when comparing alternative constructions.

For example, when Table 1 is completed by accuracies acc_1 and acc_2 computed according to Models 1 and 2, the results are

Construction	*Simplicity*	*Exactitude*	acc_0	acc_1	acc_2
DeTemple	15	10	0.002757	0.002757	0.002757
Ptolemy	16	10	0.003730	0.003949	0.003839
Labelle	15	10	0.002824	0.003289	0.003004
Hirano	19	13	0.003899	0.003914	0.003868
Richmond	45	26	0.004782	0.005990	0.005397
"Mascheroni"	34	21	0.009018	0.011593	0.009899

TABLE 12. Constructions of a regular pentagon

and practically the same with regard to goodness of various constructions.

5.5. **Computational experiment.** The relation of Model 0 to Models 1 and 2 can be illustrated by the following computational experiment[4].

Let us consider the point of intersection of two random lines L_1 and L_2, the first going through points (a, b) and $(-a, -b)$ and the second through points $(a, -b)$ and $(-a, b)$, where $a \geq b > 0$ according to Model 0. The point of intersection of lines L_1 and L_2 is $O = (0, 0)$ when $\sigma = 0$ and let $t = b/a = \tan(\alpha)$ where α is one half of the angle between lines as in (2).

An example of this this construction is displayed in Fig. 18

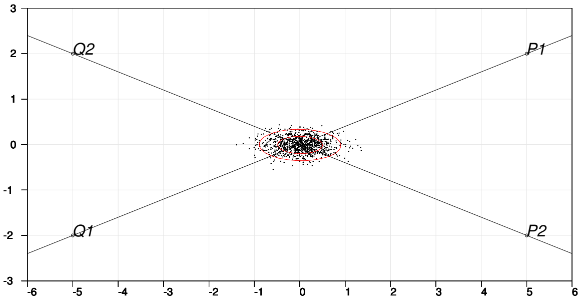


FIGURE 18

for $a = 5$, $b = 2$, illustrating also the error distribution of the point O when $\sigma = 0.3$ for 1000 replicates of the construction.

The same experiment has been repeated for a set of values t when $\sigma = 0.001$ using sample size 10^7 for each t.

By the GEOM code

```
____________________________________________________________
* a=5 b=2 c=0 STDDEV=0.001 SAMPLES=10000000
*GEOM CUR+1,E
*Cross (model 0)
*P1=point(a,b)
*Q1=point(-a,c-b)
```

[4] *Date:* 9 November 2008

```
*P2=point(a,c-b)
*Q2=point(-a,b)
*L1=line(P1,Q1)
*L2=line(P2,Q2)
*O=cross(L1,L2)
*save point(O)
E
____________________________________________________________
```

and by computations

```
____________________________________________________________
*
*FILE SAVE Geom.TXT TO NEW CR4
*CORR CROSS4,CUR+1 / VARS=OX,OY
*Means, std.devs and correlations of CR4  N=10000000
*Variable  Mean           Std.dev.
*OX       -0.000000259    0.001346577
*OY       -0.000000281    0.000538374
*Correlations:
*                  OX          OY
* OX              1.0000000  0.0004635
* OY              0.0004635  1.0000000
*
*Covariance matrix COV.M from CORR.M by
*/COV
*The eigenvalues L and eigenvectors S by
*MAT SPECTRAL DECOMPOSITION OF COV.M TO S,L
*Square roots D of eigenvalues L by
*MAT D=L
*MAT TRANSFORM D BY sqrt(X#)
*
*Square roots of eigenvalues are
*MAT_D(1)=0.00134657681521
*MAT_D(2)=0.000538374019
*
*The ratio of these numbers is
*MAT_D(2)/MAT_D(1)=0.39980936320719
*and it seems to be practically the same as t=b/a=0.4
*
*The 'relative' variance of the first principal component
*(MAT_D(1)/STDDEV)^2=1.8132691192679 (r2 in the following table)
*
*the following summary is obtained when using various values of t (or b):
*
*DATA CROSS4,a,b,n,m
D
m 1.11111111111111 1.11111 1.11111111111111  AAAAA  RR.RRRRR
n t                ratio   r2                r2'    res
a 1.00000000000000 0.99898 0.50050223979096  2/4     0.00025
* 0.80000000000000 0.79933 0.63893485173255         -0.00195
* 0.75000000000000 0.74940 0.69499747865438          0.00029
* 0.50000000000000 0.49970 1.2507024987745   5/4     0.00042
* 0.40000000000000 0.39981 1.8132691192679           0.00047
```

```
157 * 0.33333333333333 0.33320 2.5007967618061   10/4    0.00048
158 * 0.25000000000000 0.24993 4.2507345850749   17/4    0.00036
159 * 0.20000000000000 0.19996 6.5005186812805   26/4    0.00007
160 b 0.16666666666667 0.16664 9.2501475880568   37/4   -0.00039
161 *
162 *For values t of the form 1/n, the values of r2 seem to be close to ratios
163 *(n^2+1)/4=(1+t^2)/(4t^2) (compare columns r2 and r2') and this holds
164 *true also for other values as seen from the following estimation of a
165 *nonlinear regression model M done by the general estimation program
166 *ESTIMATE of Survo:
167 *
168 *a=0 b=5  / initial values in estimation
169 *MODEL M
170 *r2=(a+t^2)/(b*t^2)
171 *
172 *ESTIMATE CROSS4,M,CUR+1 / RESULTS=0
173 *Estimated parameters of model M:
174 *a=0.999050 (0.001819)
175 *b=3.996071 (0.006968)
176 *n=9 rss=0.000005 R^2=1.00000 nf=72
177 *
178 *Further computations with various values of STDDEV have indicated that
179 *the 'ratio' values have a small negative bias of magnitude 0.001 .
180 *
--------------------------------------------------------------
```

Although there is no exact proof, the above calculations indicate that when using MODEL=0 and STDDEV=σ, in the distribution of point O, the eigenvalues of the covariance matrix are (save for a slight bias)

$$\lambda_1 = C/t^2,$$
$$\lambda_2 = C$$

where

$$C = \sigma^2(1+t^2)/4$$

and the eigenvectors form an identity matrix. The ratio of eigenvalues is

$$\lambda_1/\lambda_2 = 1/t^2 = 1/\tan^2\alpha$$

i.e. the same as in conditional error distribution of O according to Models 1 and 2.

λ_1 is also equal to the variance of the X coordinate of O and similarly, λ_2 is equal to the variance of the Y coordinate of O.

These results seem to be valid for any combination of a and b depending only on their ratio $t = b/a$ at least when $a \geq b > 100\sigma$.

Hence this experiment clarifies why Model 0 gives similar results as Models 1 and 2. One can say that the conditional prior distribution of a point in Model 0 corresponds to conditional posterior distribution in Models 1 and 2.

6. GEOM program for computing accuracies of constructions

Although there are plenty of programs for making geometric constructions [5] none of them permit calculations needed in this context. Therefore a new program was made for
a. describing constructions,
b. plotting graphs of constructions, and
c. calculating statistical accuracies of constructions by simulation.

This program has been written in C as a program module of the SURVO MM system and it is activated by a `GEOM` command in this environment. As all commands in Survo, `GEOM` gets its main input from the current edit field containing both a sequence of construction commands and various extra specifications for control of the current task.

`GEOM` is used in two ways. By default, it only interprets the construction commands and saves various geometric objects thus generated in specific Survo data files. The points are saved in `_Points.SVO`, lines in `_Lines.SVO`, circles in `_Circles.SVO`, and line segments in `_Edges.SVO`.

`GEOM` does not make any graphs. They are drawn by various `PLOT` commands of Survo by using the above mentioned Survo data files as input. The `PLOT` commands needed - with specifications of their own - are readily available in the environment (edit field) where `GEOM` is typically used and they are simultaneusly activated - by an `/ACTIVATE` command - producing a graph of the construction as a PostScript file `_Geom.PS`. This file is converted into a PDF file (for example, by a `/GS-PDF` command of Survo) and displayed thereafter in its own window.

When making a construction by these tools, it is easy to proceed stepwise by typing the `GEOM` code and seeing the graph produced by current code immediately.

The second and more important usage of `GEOM` takes place when a `SAMPLES=`n specification is present. Then the construction is repeated n times using an error model given by `MODEL` and nominal accuracy given by `STDDEV`. The files `_Points.SVO`, `_Lines.SVO`, etc. are generated for each replicate and pertinent information given by save commands in `GEOM` code is saved as an ascii file `Geom.TXT`. This file now containing n cases of the sampled construction is easily converted into a Survo data file and this data file is used for making statistical summaries about the accuracy of the construction by using standard tools of the Survo system.

Thus the GEOM program as such is limited to the essential task for defining the construction in presence of possible measurement error. The standard operations of Survo take care of all other activities needed.

The GEOM program is available in Survo in versions 3.01+. An edit field (file) `_GEOM` works as a template for constructions. By default, it contains DeTemple's construction for a regular pentagon.

Any other construction can be created by editing this template in Survo and by saving it as a new edit file with a suitable name.

```
____________________________________________________________
*SAVE _GEOM / Template for GEOM applications
*
**GLOBAL* specifications for plotting
*
*X1=-2 X2=1 Y1=-2 Y2=2  Limits of the plotting area
```

```
*s=250                  Unit length
*XSCALE=X1(1)X2 SIZE=(X2-X1+1)*s,(Y2-Y1+1)*s
*YSCALE=Y1(1)Y2 XDIV=0.5,X2-X1,0.5 YDIV=0.5,Y2-Y1,0.5
*HEADER=
*FRAME=1
*XLABEL=  YLABEL=
*HOME=0,0
*
*......................................................................
*Sampling specifications:
*RAND=2008 SAMPLES=1000000 STDDEV=0.001   MODEL=0
*................... Open this border for sampling
*
*This /ACTIVATE makes the graph after /GEOM below has been activated:
*/ACTIVATE +
*
*Sucro /GEOM runs the GEOM code, creates files for geometric objects
*and, when SAMPLES>0, saves simulation results in Geom.TXT.
*GOTO X / Working with Geom.TXT
*
*/GEOM
*GEOM CUR+1,E,E+1
*Pentagon (DeTemple)
*O=point(0,0)
*Q=point(-1,0)
*_C1=circle(O,1)
*LY0=line(O,_Q)
*P0=cross_cl(C1,LY0,1,0)
*LX0=perpendicular(LY0,*O)
*A=cross_cl(C1,LX0,0,1)
*
*symbol start
*M=midpoint(*O,Q,LY0)
*C5=circle_p(M,A)
*H0=cross_cl(C5,LY0,1,0)
*H1=cross_cl(C5,LY0,-1.5,0)
*C6=circle(H1,1,-0.25,0.25)
*P2=cross_cc(C1,C6,-1,0.5)
*P3=cross_cc(C1,C6,-1,-0.5)
*
*C7=circle(H0,*1,0.5,1.5)
*P1=cross_cc(C1,C7,0,1)
*P4=cross_cc(C1,C7,0,-1)
*
*E=edge(P0,P1)
*E1=edge(P1,P2)
*E2=edge(P2,P3)
*E3=edge(P3,P4)
*E4=edge(P4,P0)
*save edge(E)
*save point(P0)
*save point(P1)
*save point(P2)
```

```
*save point(P3)
*save point(P4)
E
*GEOM construction: Pentagon (DeTemple)
*Geometrographic symbol = 2S1+1S2+8C1+0C2+4C3
*Simplicity = 15  Exactitude = 10
*
*......................................................................
*Points created in the accurate construction:
*
*FILE LOAD _Points / VARS=Point,X,Y
*DATA _Points*,A,B,C
C Point                               X                    Y
A O                  0.0000000000000000   0.0000000000000000
* Q                 -1.0000000000000000   0.0000000000000000
* P0                 1.0000000000000000   0.0000000000000000
* A                  0.0000000000000001   1.0000000000000000
* M                 -0.5000000000000000   0.0000000000000000
* H0                 0.6180339887498949   0.0000000000000000
* H1                -1.6180339887498949   0.0000000000000000
* P2                -0.8090169943749475   0.5877852522924731
* P3                -0.8090169943749475  -0.5877852522924731
* P1                 0.3090169943749475   0.9510565162951535
B P4                 0.3090169943749475  -0.9510565162951535
*
*
* P0            p0X=+1.0000000000000000  p0Y=+0.0000000000000000
* P1            p1X=+0.3090169943749475  p1Y=+0.9510565162951535
* P2            p2X=-0.8090169943749475  p2Y=+0.5877852522924731
* P3            p3X=-0.8090169943749475  p3Y=-0.5877852522924731
* P4            p4X=+0.3090169943749475  p4Y=-0.9510565162951535
*
*Checking by editorial computing (using trigonometric expressions):
*pi=3.141592653589793
*             X(I):=cos(I*2*pi/5)      Y(I):=sin(I*2*pi/5)
*
*             X(0).=1                  Y(0).=0
*             X(1).=0.30901699437495   Y(1).=0.95105651629515
*             X(2).=-0.80901699437495  Y(2).=0.58778525229247
*             X(3).=-0.80901699437495  Y(3).=-0.58778525229247
*             X(4).=0.30901699437495   Y(4).=-0.95105651629515
*
*......................................................................
*Default PLOT schemes for GEOM (activated automatically by /ACTIVATE)
*
*GRID=[color(0.1,0.1,0.1,0)],1,1
+PLOT X(t)=x0+r*cos(u*t+v),Y(t)=y0+r*sin(u*t+v)
*DEVICE=PS,_Circles.PS
*t=[RED],0(pi/60)2*pi  pi=3.141592653589793
*x0=DATA:_Circles,X
*y0=DATA:_Circles,Y
*r=DATA:_Circles,R
*u=DATA:_Circles,A1
```

```
*v=DATA:_Circles,A2
*.....................................................................
*t=-10,10,1
+PLOT X(t)=x1+t*(x2-x1),Y(t)=y1+t*(y2-y1)
*DEVICE=PS,_Lines.PS
*x1=DATA:_Lines,X1
*y1=DATA:_Lines,Y1
*x2=DATA:_Lines,X2
*y2=DATA:_Lines,Y2
*.....................................................................
*t=[GREEN][line_width(1)],0,1,1
+PLOT X(t)=x1+t*(x2-x1),Y(t)=y1+t*(y2-y1)
*DEVICE=PS,_Edges.PS
*x1=DATA:_Edges,X1
*y1=DATA:_Edges,Y1
*x2=DATA:_Edges,X2
*y2=DATA:_Edges,Y2
*.....................................................................
+PLOT _Points,X,Y
*DEVICE=PS,_Points.PS
*POINT=3,5
*............................
+PLOT _Points,X,Y
*DEVICE=PS,_PointsN.PS
*POINT=[move(15,15)][Swiss0(7)],Point
*.....................................................................
*
*Combining graphs (PS files) of various geometric objects:
+EPS JOIN _GeomN,_Circles,_Lines,_Edges,_Points,_PointsN
*
*Displaying the final graph in its own window:
+/GS-PDF _GeomN.PS
*
*   (End of automatically activated commands)
*
*.....................................................................
X
*Working with samples in Geom.TXT:
*
*Conversion to a Survo data file:
*FILE SAVE Geom.TXT TO NEW K
*
*Constants related to accurate vertices (algebraic expressions):
*     f=(sqrt(5)+1)/2   "golden ratio"
*     r1x=1/2*(f-1)
*     r1y=1/2*sqrt(2+f)
*     r2x=-f/2
*     r2y=1/2*sqrt(3-f)
*     edge length: sqrt(3-f)=1.1755705045849
*
* Coordinates of vertices    Squared errors of vertices
* P0: p0X=1    p0Y=0         d0=(P0X-p0X)^2+(P0Y-p0Y)^2
* P1: p1X=r1x  p1Y=r1y       d1=(P1X-p1X)^2+(P1Y-p1Y)^2
```

```
165 * P2: p2X=r2x  p2Y=r2y       d2=(P2X-p2X)^2+(P2Y-p2Y)^2
166 * P3: p3X=r2x  p3Y=-r2y      d3=(P3X-p3X)^2+(P3Y-p3Y)^2
167 * P4: p4X=r1x  p4Y=-r1y      d4=(P4X-p4X)^2+(P4Y-p4Y)^2
168 *
169 *Computing total squared error:
170 *VAR D2:8=d0+d1+d2+d3+d4 TO K
171 *
172 *MINSTAT K,CUR+1 / VARS=E,D2
173 *Basic statistics of data K N=1000000
174 *Variable   mean         stddev             N  minimum      maximum
175 * E         1.175573756  0.002266360  1000000  1.165463791  1.186834877
176 * D2        0.000037995  0.000037299  1000000  0.000000036  0.000578814
177 *
178 *Accuracy acc0:
179 *sqrt(MAT_STAT.M(D2,mean)/5)=0.00275663698572
------------------------------------------------------------
```

Especially for those who are not familiar with the *editorial interface* of Survo, the following explanations may be helpful.

The preceding setup of 179 lines is an extract of an *edit field* of Survo. When working with Survo, always a portion of the edit field (typically 23 or more lines and 72 or more columns) is displayed in a window of the computer screen. The user can type and edit text in this environment as in a word processing program, but it is also possible to type commands and *activate* them (typically by double-clicks of the mouse). Any line of the edit field can be a command line. The results of commands appear in the same edit field as text or they (for example, numerical tables and matrices) are saved in files. Similarly, results in pictorial mode (like curves or statistical graphs) are saved in files (either in PostScript or Enhanced MetaFile format) and they are displayed in separate windows, too.

Commands may have parameters typed after the command word but more complicated commands employ also *specifications* located somewhere in the edit field. For example, in the _GEOM edit field, the first lines (1–14)

```
-------------------------------------------------------------
  1 *SAVE _GEOM / Template for GEOM applications
  2 *
  3 **GLOBAL* specifications for plotting
  4 *
  5 *X1=-2 X2=1 Y1=-2 Y2=2  Limits of the plotting area
  6 *s=250                  Unit length
  7 *XSCALE=X1(1)X2 SIZE=(X2-X1+1)*s,(Y2-Y1+1)*s
  8 *YSCALE=Y1(1)Y2 XDIV=0.5,X2-X1,0.5 YDIV=0.5,Y2-Y1,0.5
  9 *HEADER=
 10 *FRAME=1
 11 *XLABEL=  YLABEL=
 12 *HOME=0,0
 13 *
 14 *......................................................................
------------------------------------------------------------
```

contain *global* settings for all commands in the same edit field. The text `*GLOBAL*` (on line 3) tells that specifications given before the first *border line* (line 14, consisting of plain '.'s) are valid for any later commands. In this case, all these specifications are related to plotting of graphs (by `PLOT` commands later on lines 105, 115, etc.) For example, the line 7 gives (by `XSCALE`) the scale notations -2, -1, 0, 1 for the X axis of the graph and (by `SIZE`) the size of the graph as 1000×1250 in plotting units. Both of these two specifications depend on parameter values given on lines 5 and 6. Thus the specifications may depend on other specifications and usually they can be presented in any order in the edit field. The only command on these first 14 lines is `SAVE _GEOM`. When it is activated, the entire contents of the edit field is saved in a (text) file `_GEOM.EDT`. In practice, when creating and studying a new construction, the filename should be changed so that it describes the new project.

The first task of the user is to replace the GEOM code (now on lines 27–61) by a new one. It is usually best to write code from the beginning in suitable parts and test its validity stepwise.

In each step, the code is activated by the `/GEOM` command here on line 26.

```
------------------------------------------------------------
 14 *..................................................................
 15 *Sampling specifications:
 16 *RAND=2008 SAMPLES=1000000 STDDEV=0.001   MODEL=0
 17 *................... Open this border for sampling
 18 *
 19 *This /ACTIVATE makes the graph after /GEOM below has been activated:
 20 */ACTIVATE +
 21 *
 22 *Sucro /GEOM runs the GEOM code, creates files for geometric objects
 23 *and, when SAMPLES>0, saves simulation results in Geom.TXT.
 24 *GOTO X / Working with Geom.TXT
 25 *
 26 */GEOM
 27 *GEOM CUR+1,E,E+1
 28 *Pentagon (DeTemple)
 29 *O=point(0,0)
 30 *Q=point(-1,0)
------------------------------------------------------------
```

The slash '/' in front of the command indicates that `/GEOM` is a *sucro* command activating a **Su**rvo ma**cro** command written in the macro code of Survo. This sucro merely creates files for various objects (points, lines, circles, etc.) generated in the geometric construction and finally activates the command (`GEOM`) on the next line (27). The `GEOM CUR+1,E,E+1` command calls the GEOM program to read the code of the construction on lines from `CUR+1` (current line + 1 = 28) to `E` (61). The last parameter `E+1` tells that the summary of geometrographic characteristics is to placed from this line (62) onwards and there it is (lines 62–64). It is better to use symbolic line labels like `E` instead of their numeric counterparts like 61 since, for example, if some lines above `E` are inserted or deleted, the symbolic label is correct but the numerical label is not. Symbolic labels (typically letters) appear in the *control column* in front of the proper edit line.

When the GEOM program starts, its behaviour depends on the `SAMPLES` specification here on line 16. But in the current situation, it is a local specification and valid only between the border lines 14–17. Therefore the GEOM program cannot 'see' any `SAMPLES` specification and then its only task is to create files describing the 'accurate' construction according to the GEOM code on lines 27–61.

The geometric objects created in the construction are saved in Survo data files `_Points.SVO` (points), `_Lines.SVO` (straight lines), `_Circles` (circles or arcs of circles), `_Edges` (line segments).

For example, the points appearing in the construction have been loaded from file `_Points.SVO` by a `FILE LOAD` command

```
____________________________________________________________
 67 *Points created in the accurate construction:
 68 *
 69 *FILE LOAD _Points / VARS=Point,X,Y
 70 *DATA _Points*,A,B,C
 71 C Point                                X                   Y
 72 A O                  0.0000000000000000   0.0000000000000000
 73 * Q                 -1.0000000000000000   0.0000000000000000
 74 * P0                 1.0000000000000000   0.0000000000000000
 75 * A                  0.0000000000000001   1.0000000000000000
 76 * M                 -0.5000000000000000   0.0000000000000000
 77 * H0                 0.6180339887498949   0.0000000000000000
 78 * H1                -1.6180339887498949   0.0000000000000000
 79 * P2                -0.8090169943749475   0.5877852522924731
 80 * P3                -0.8090169943749475  -0.5877852522924731
 81 * P1                 0.3090169943749475   0.9510565162951535
 82 B P4                 0.3090169943749475  -0.9510565162951535
___________________________________________________________
```

The coordinates of the most important points, the vertices of the pentagon have been copied from this list and their correctness is checked by using the corresponding trigonometric expressions:

```
____________________________________________________________
 85 * P0           p0X=+1.0000000000000000  p0Y=+0.0000000000000000
 86 * P1           p1X=+0.3090169943749475  p1Y=+0.9510565162951535
 87 * P2           p2X=-0.8090169943749475  p2Y=+0.5877852522924731
 88 * P3           p3X=-0.8090169943749475  p3Y=-0.5877852522924731
 89 * P4           p4X=+0.3090169943749475  p4Y=-0.9510565162951535
 90 *
 91 *Checking by editorial computing (using trigonometric expressions):
 92 *pi=3.141592653589793
 93 *            X(I):=cos(I*2*pi/5)      Y(I):=sin(I*2*pi/5)
 94 *
 95 *            X(0).=1                  Y(0).=0
 96 *            X(1).=0.30901699437495   Y(1).=0.95105651629515
 97 *            X(2).=-0.80901699437495  Y(2).=0.58778525229247
 98 *            X(3).=-0.80901699437495  Y(3).=-0.58778525229247
 99 *            X(4).=0.30901699437495   Y(4).=-0.95105651629515
___________________________________________________________
```

On line 93, general expressions of for the coordinates of the regular pentagon are defined as *temporary* functions. Now, by activating any of the items on lines 95–99 produces values for them all. This is a typical application of *editorial computing* in Survo. The correspondence of results is easy to verify by eye, but it can also be done by activating expressions like `Y(4)-p4Y=` .

The /GEOM command creates only numerical results describing the results of the construction. When the actual graph is desired, then the sucro command /ACTIVATE + (on line 20) makes the job. This is a standard sucro for activating a sequence of commands in the current edit field. In this case, the parameter + states that all lines having a + in the control column are to be activated. In this template, these lines appear here:

```
------------------------------------------------------------
101 *.....................................................................
102 *Default PLOT schemes for GEOM (activated automatically by /ACTIVATE)
103 *
104 *GRID=[color(0.1,0.1,0.1,0)],1,1
105 +PLOT X(t)=x0+r*cos(u*t+v),Y(t)=y0+r*sin(u*t+v)
106 *DEVICE=PS,_Circles.PS
107 *t=[RED],0(pi/60)2*pi  pi=3.141592653589793
108 *x0=DATA:_Circles,X
109 *y0=DATA:_Circles,Y
110 *r=DATA:_Circles,R
111 *u=DATA:_Circles,A1
112 *v=DATA:_Circles,A2
113 *.....................................................................
114 *t=-10,10,1
115 +PLOT X(t)=x1+t*(x2-x1),Y(t)=y1+t*(y2-y1)
116 *DEVICE=PS,_Lines.PS
117 *x1=DATA:_Lines,X1
118 *y1=DATA:_Lines,Y1
119 *x2=DATA:_Lines,X2
120 *y2=DATA:_Lines,Y2
121 *.....................................................................
122 *t=[GREEN][line_width(1)],0,1,1
123 +PLOT X(t)=x1+t*(x2-x1),Y(t)=y1+t*(y2-y1)
124 *DEVICE=PS,_Edges.PS
125 *x1=DATA:_Edges,X1
126 *y1=DATA:_Edges,Y1
127 *x2=DATA:_Edges,X2
128 *y2=DATA:_Edges,Y2
129 *.....................................................................
130 +PLOT _Points,X,Y
131 *DEVICE=PS,_Points.PS
132 *POINT=3,5
133 *............................
134 +PLOT _Points,X,Y
135 *DEVICE=PS,_PointsN.PS
136 *POINT=[move(15,15)][Swiss0(7)],Point
137 *.....................................................................
138 *
139 *Combining graphs (PS files) of various geometric objects:
140 +EPS JOIN _GeomN,_Circles,_Lines,_Edges,_Points,_PointsN
141 *
142 *Displaying the final graph in its own window:
143 +/GS-PDF _GeomN.PS
144 *
-----------------------------------------------------------
```

This segment consists of some plotting *schemes* i.e. `PLOT` commands with several local specifications.

The simplest of them locates on lines 130–132. There the `PLOT` command (now automatically activated) makes a scatter plot of variables `X,Y` in the data file `_Points.SVO`. The `DEVICE` specification (131) tells that the graph will be saved as PostScript code in file `_Points.PS` and `POINT=3,5` specifies the type and the size of the point image in the graph. The size and other general properties of the graph are selected by the global specifications in the first lines of the edit field.

The `_Points.SVO` file also has the labels of points as the first variable `Point` (see line 71) and they are plotted by the `PLOT` command on line 134.

The straight lines drawn in the construction are saved in `_Lines.PS` by the plotting scheme on lines 114–120. The general equation of a straight line is described in parametric form on the `PLOT` line 115 and the points `(x1,y1)`, `(x2,y2)` determining each line are taken from the data file `_Lines.SVO` by specifications on lines 117–120. Since the lines are drawn by funtions of `t`, a suitable range and step length is given on line 114.

The plotting scheme for line segments (lines 122–128) are similar to that of straight lines. Now the range (0,1) for `t` is obvious. In the `t` specification (122) the color is set 'green' and the line width 1, i.e line segments are drawn as green thick objects.

In the scheme for circles and arcs of them (lines 104–112) again a standard parametric form (polar coordinates) is used but in a generalized form (constants `u` and `v`). These parameters enable both complete circles and parts of them to be drawn.

It is easy to alter the style used in actual plotting by changing specifications in this section. The current setup creates just the graph on the left side of Fig. 2.

The final graph is obtained by combining the PostScript files thus generated by the `EPS JOIN` command on line 140. The user can decide here what files are included. Sometimes also new files (like those for plotting scatter diagrams on the right side of Fig. 2) may be added to the list.

The combined file is displayed on the screen by the command `/GS-PDF` on line 143. This is a general sucro command that converts a PostScript file into a PDF format by using the Ghostscript program and then makes the display by Adobe Acrobat Reader.

In this conncetion, the second alternative, sampling according to some error model, is more important. Then the border line 17 is removed and when the `/GEOM` command on line 26 is activated, the GEOM program is able to detect the additional specifications on line 16.

The setting `RAND=2008 SAMPLES=1000000 STDDEV=0.001 MODEL=0` implies that one million samples of the construction will be generated according to Model 0 and using $\sigma = 0.001$. Thus the GEOM code is repeated 10^6 times and in each repetition the points generated are randomized. As a consequence, all other geometric objects will be randomized, too. In each repetition files `_Points.SVO`, `_Lines.SVO`, etc. are regenerated with the randomized components and now the `save` commands on lines 55–60 take care of saving selected items in `Geom.TXT`.

Within a few seconds (when sample size is one million) the process is completed and thereafter the `GOTO X` command on line 24 leads us to a section:

```
-------------------------------------------------------
```

```
148 X
149 *Working with samples in Geom.TXT:
150 *
151 *Conversion to a Survo data file:
152 *FILE SAVE Geom.TXT TO NEW K
153 *
154 *Constants related to accurate vertices (algebraic expressions):
155 *     f=(sqrt(5)+1)/2   "golden ratio"
156 *     r1x=1/2*(f-1)
157 *     r1y=1/2*sqrt(2+f)
158 *     r2x=-f/2
159 *     r2y=1/2*sqrt(3-f)
160 *     edge length: sqrt(3-f)=1.1755705045849
161 *
162 * Coordinates of vertices    Squared errors of vertices
163 * P0: p0X=1    p0Y=0         d0=(P0X-p0X)^2+(P0Y-p0Y)^2
164 * P1: p1X=r1x  p1Y=r1y       d1=(P1X-p1X)^2+(P1Y-p1Y)^2
165 * P2: p2X=r2x  p2Y=r2y       d2=(P2X-p2X)^2+(P2Y-p2Y)^2
166 * P3: p3X=r2x  p3Y=-r2y      d3=(P3X-p3X)^2+(P3Y-p3Y)^2
167 * P4: p4X=r1x  p4Y=-r1y      d4=(P4X-p4X)^2+(P4Y-p4Y)^2
168 *
169 *Computing total squared error:
170 *VAR D2:8=d0+d1+d2+d3+d4 TO K
171 *
172 *MINSTAT K,CUR+1 / VARS=E,D2
173 *Basic statistics of data K N=1000000
174 *Variable   mean         stddev             N  minimum      maximum
175 * E         1.175573756  0.002266360  1000000  1.165463791  1.186834877
176 * D2        0.000037995  0.000037299  1000000  0.000000036  0.000578814
177 *
178 *Accuracy acc0:
179 *sqrt(MAT_STAT.M(D2,mean)/5)=0.00275663698572
------------------------------------------------------------
```

At first, the text file Geom.TXT is converted to a Survo data file K.SVO and it inherits besides the sample of selected objects also their names used in the construction.

What follows after that, depends essentially on the nature of the construction. In this construction, it is appropriate to consider, how accurately the side length E of the pentagon is estimated and as the most important thing, how accurately the vertices of the pentagon are obtained. In the latter case, the total sum of squares D2 of the deviations of the X and Y coordinates from the true ones is a good overall measure. Then the square root of the mean squared error divided by 5 gives the total error reduced to the scale of the nominal error σ. It is denoted acc0 and calculated as follows.

At first, the true coordinates of vertices are expressed as functions of the golden ratio f (see lines 154–167). The true values were given already in trigonometric form on lines 85–89, but here it is good to emphasize the relation of the regular pentagon to the golden ratio.

In fact, I didn't derive these formulas exactly. I merely applied inverse symbolic computing by using the INTREL command of Survo. It is based on the PSLQ algorithm, see [4], [19].

For example, the expression for p1X is guessed as follows:

```
____________________________________________________________
The numerical value of p1X from _Points.SVO is
p1X=+0.3090169943749475
INTREL p1X,CUR+1 / DEGREE=2
X=p1X is a root of 4*X^2+2*X-1=0  (result of INTREL)
Hence the positive root of this equation is
X=(-2+sqrt(4+16)/8=(-1+sqrt(5))/4=(f-1)/2=r1X=p1X
___________________________________________________________
```

The values of variable D2 are computed by the VAR command on line 170. Notation D2:8 tells that values of this new variable should be saved in double precision (taking 8 bytes).

A summary of basic statistics for variables E and D2 is produced by the MINSTAT command on line 172 giving the results as text on lines 173–176, but they are simultaneously saved in double precision in a predetermined matrix file STAT.M.

The scaled RMSE of construction of the vertices can then be computed as a matrix expression on line 179 showing that the mean accuracy of the construction for one vertex is 0.0028 or 2.8 times the nominal accuracy.

In the preceding excerpts from a Survo edit field, the commands activated by the user are colored in red (in order to increase readability). When using Survo, it is typical to employ different color combinations for miscellaneous purposes. In fact, any line in the edit field can have a *shadow line* for specifying *shadow characters*. A shadow line can be revealed for editing by a certain key combination and in some situations, shadow characters have real importance. For example, shadow '1' displays a character in red and when printing a text, such characters appear in boldface by default.

7. Supplement 1: "Finding two needles in a haystack"

My interest in this topic arose when a chief shop steward of a big Finnish firm called me (in January 2008) and claimed that he is able to "square a circle with a compass and ruler". I tried to explain him that it has been known over 100 years that it is an impossible task. He didn't believe and sent me a short construction which proved to be an approximate solution with a 4 per cent error.

It is told, for example, in `http://en.wikipedia.org/wiki/Circle_Squaring` that

> "Though squaring the circle is an impossible problem using only compass and straightedge, approximations to squaring the circle can be given by constructing lengths close to π. It takes only minimal knowledge of elementary geometry to convert any given rational approximation of π into a corresponding compass-and-straightedge construction, but constructions made in this way tend to be very long-winded in comparison to the accuracy they achieve. After the exact problem was proven unsolvable, some mathematicians have applied their ingenuity to finding elegant approximations to squaring the circle, defined roughly (and informally) as constructions that are particularly simple among other imaginable constructions that give similar precision."

I decided to try find good approximate solutions by a random search in an $n \times n$ grid as follows. I made a C program as a Survo module for sampling points (x, y) for $x, y = 0, 1, 2, \ldots, n$ and connecting pairs of points with straight lines and storing 3000 points of intersection (u, v) of such random lines when $0 \leq u, v \leq n$.

Thereafter all squared distances of points (u, v) were computed and sorted in ascending order. After cancelling too short distances (less than 4 in case $n = 16$) it was easy to find "the needles", i.e. the ratio of the squared distances closest to π. One such experiment took about half a minute on my PC.

The experiment was repeated several thousand times and the best of them gave a rational approximation

2439604544650/776550244941=3.1415926535897931067 …

which deviates from π by 1.3×10^{-16}.

I told about this result in the Finnish newsgroup of Survo users and presented the solution as a graph

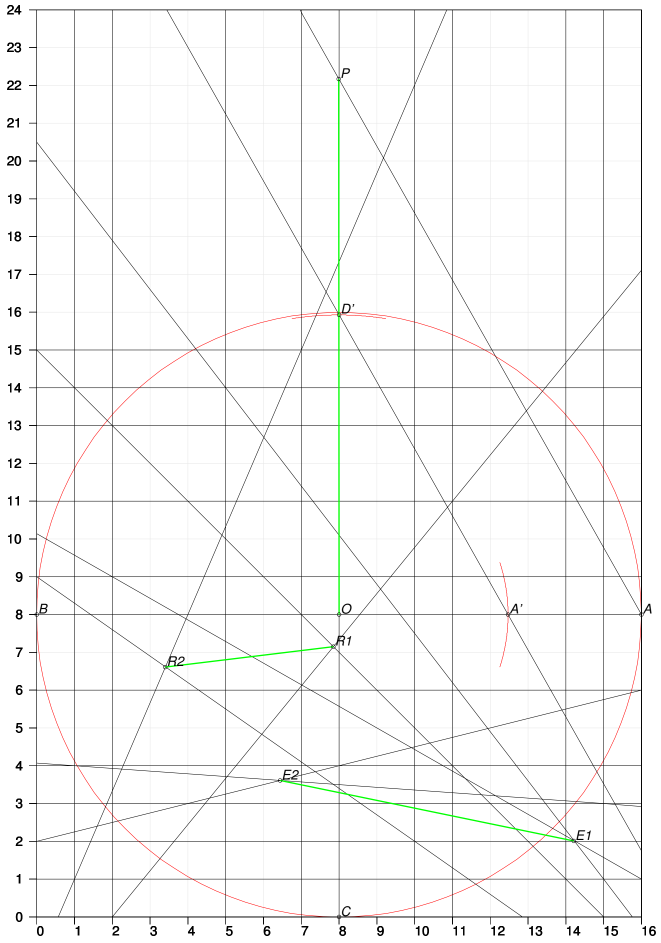
`http://www.survo.fi/papers/p34838.pdf`

plotted by Survo.

The solution is based on the points

	X_{11}	Y_{11}	X_{12}	Y_{12}	X_{21}	Y_{21}	X_{22}	Y_{22}
R_1	14	1	2	13	2	0	11	11
R_2	10	2	0	9	1	1	7	15
E_1	16	1	2	9	15	1	5	14
E_2	15	3	1	4	0	2	16	6

so that, for example, the first end point R_1 of the edge corresponding to the radius of the circle is the point of intersection of a line from $(14, 1)$ to $(2, 13)$ and a line from $(2, 0)$ to $(11, 11)$. Similarly, the second end point of the radius is described in the table on line R_2, and the end points of the edge of the square on lines E_1 and E_2.



The final construction is based on a circle of radius 8 in the 16×16 grid. At the end, the original edges have to be transferred so that the $OA' = R_1R_2$ and $OD' = E_1E_2$ and the parallel to line $A'D'$ going through A intersects line OD' in point P giving the edge OP of the square corresponding to the original circle of radius $OA = 8$.

The following GEOM code describes the entire construction.

```
------------------------------------------------------------
*GEOM CUR+1,E
*Grid16 construction
*O=point(8,8)
*A=point(16,8)
*_C1=circle(O,8)
*LY8=line(_A,O)
*B=cross_cl(C1,LY8,0,8)
*LX8=perpendicular(LY8,O)
*C=cross_cl(C1,LX8,8,0)
*d=cross_cl(C1,LX8,8,16)
*p4_8=midpoint(B,O,LY8)
*p2_8=midpoint(B,P4_8,LY8)
```

```
*LX2=perpendicular(LY8,P2_8)
*p1_8=midpoint(B,P2_8,LY8)
*LX1=perpendicular(LY8,P1_8)
*LX0=perpendicular(LY8,B)
*p6_8=midpoint(P4_8,O,LY8)
*p5_8=midpoint(P4_8,P6_8,LY8)
*LX5=perpendicular(LY8,P5_8)
*p7_8=midpoint(P6_8,O,LY8)
*LX7=perpendicular(LY8,P7_8)
*p12_8=midpoint(O,A,LY8)
*p10_8=midpoint(O,P12_8,LY8)
*LX10=perpendicular(LY8,P10_8)
*p11_8=midpoint(P10_8,P12_8,LY8)
*LX11=perpendicular(LY8,P11_8)
*p14_8=midpoint(P12_8,A,LY8)
*LX14=perpendicular(LY8,P14_8)
*p15_8=midpoint(P14_8,A,LY8)
*LX15=perpendicular(LY8,P15_8)
*LX16=perpendicular(LY8,A)
*
*p8_4=midpoint(O,C,LX8)
*LY4=perpendicular(LX8,P8_4)
*p8_2=midpoint(C,P8_4,LX8)
*LY2=perpendicular(LX8,P8_2)
*p8_1=midpoint(C,P8_2,LX8)
*LY1=perpendicular(LX8,P8_1)
*LY0=perpendicular(LX8,C)
*p8_3=midpoint(P8_2,P8_4,LX8)
*LY3=perpendicular(LX8,P8_3)
*p8_6=midpoint(P8_4,O,LX8)
*LY6=perpendicular(LX8,P8_6)
*p8_12=midpoint(O,D,LX8)
*p8_10=midpoint(O,P8_12,LX8)
*p8_9=midpoint(O,P8_10,LX8)
*LY9=perpendicular(LX8,P8_9)
*p8_11=midpoint(P8_12,P8_10,LX8)
*LY11=perpendicular(LX8,P8_11)
*p8_14=midpoint(P8_12,D,LX8)
*LY14=perpendicular(LX8,P8_14)
*p8_13=midpoint(P8_12,P8_14,LX8)
*LY13=perpendicular(LX8,P8_13)
*p8_15=midpoint(P8_14,D,LX8)
*LY15=perpendicular(LX8,P8_15)
*
*p0_2=cross(LX0,LY2)
*p0_9=cross(LX0,LY9)
*p1_1=cross(LX1,LY1)
*p1_4=cross(LX1,LY4)
*p2_0=cross(LX2,LY0)
*p2_9=cross(LX2,LY9)
*p2_13=cross(LX2,LY13)
*p5_14=cross(LX5,LY14)
*p7_15=cross(LX7,LY15)
```

```
166 *p10_2=cross(LX10,LY2)
167 *p11_11=cross(LX11,LY11)
168 *p14_1=cross(LX14,LY1)
169 *p15_1=cross(LX15,LY1)
170 *p15_3=cross(LX15,LY3)
171 *p16_1=cross(LX16,LY1)
172 *p16_6=cross(LX16,LY6)
173 *LR11=line(P14_1,P2_13)
174 *LR12=line(P2_0,P11_11)
175 *R1=cross(LR11,LR12)
176 *LR21=line(P10_2,P0_9)
177 *LR22=line(P1_1,P7_15)
178 *R2=cross(LR21,LR22)
179 *R=edge(R1,R2)
180 *
181 *LS11=line(P16_1,P2_9)
182 *LS12=line(P15_1,P5_14)
183 *E1=cross(LS11,LS12)
184 *LS21=line(P15_3,P1_4)
185 *LS22=line(P0_2,P16_6)
186 *E2=cross(LS21,LS22)
187 *S=edge(E1,E2)
188 *
189 *C2=circle(O,R,-0.1,0.1)
190 *A'=cross_cl(C2,LY8,13,8)
191 *C3=circle(O,S,0.45,0.55)
192 *D'=cross_cl(C3,LX8,8,16)
193 *L'=line(A',D')
194 *LAP=parallel(L',A)
195 *P=cross(LAP,LX8)
196 *Edge=edge(O,P)
197 *save edge(Edge)
198 E
------------------------------------------------------------
```

The statistical accuracy according to Model 0 for various nominal accuracies is

σ_r	RMSE
10^{-2}	15.437
10^{-3}	15.284
10^{-4}	15.282
10^{-5}	15.282
10^{-6}	15.282
10^{-7}	15.282
10^{-8}	15.282
10^{-9}	15.282
10^{-10}	15.289
10^{-11}	15.321
10^{-12}	15.321
10^{-13}	15.322
10^{-14}	15.520
10^{-15}	28.665

TABLE 13. Accuracy of the Grid16 construction for various nominal errors

Thus Ramanujan's second construction is almost four times more accurate for $\sigma_r > 10^{-9}$ but the Grid16 construction beats it for $\sigma_r < 10^{-11}$. However, the Vieta-Osler construction beats them both.

Of course, results for very high nominal accuracies have no practical value, but they still describe goodness of methods for "approximate circle squaring" in a theoretical sense.

The weakness of the Grid16 construction is that the solution is obtained as a ratio estimate (i.e. ratio of of two random variables) and not as a ratio of random variable (edge of square) and a constant (radius of the original circle) as in constructions in Section 4.

Since this Grid16 construction was achieved by a random search within a limited time span, still better alternatives may exist. However, for the smallest grid sizes $n = 2, 3, 4$ the best possible solutions can be found and they are

n	ratio	'π'	bias
2	41/13	3.15...	1.2×10^{-2}
3	22321/7105	3.141590...	2.2×10^{-6}
4	5311453/1690688	3.1415926534...	1.8×10^{-10}

All possible points of intersections (u, v) of the lines from one grid point to another are displayed in the following figure.

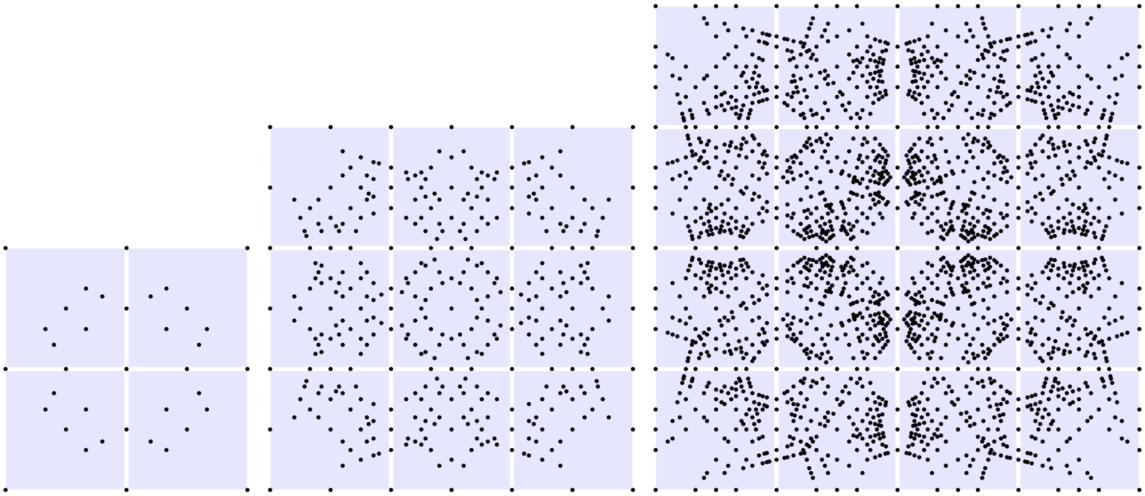
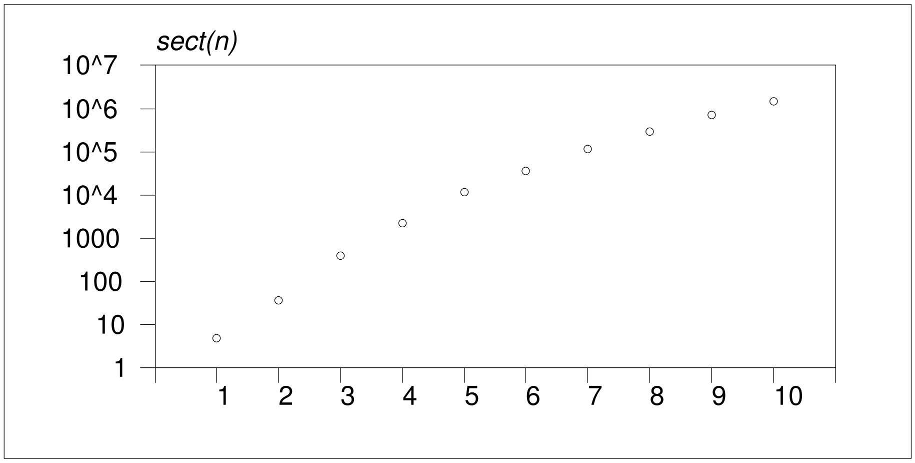


FIGURE 19. Points of intersections (u, v) for $n = 2, 3, 4$

The numbers $sect(n)$ of distinct (u, v) points for $n = 1, 2, \ldots, 10$ are

n	1	2	3	4	5	6	7	8	9	10
$sect(n)$	5	37	405	2225	11641	35677	114409	295701	718469	1475709

These numbers have been computed by brute force. The problem of finding a general expression for $sect(n)$ remains open.



When values of *sect*(n) are plotted in logarithmic scale against n, it can be seen that the growth is fast but obviously polynomial, not exponential. The values seem to be slightly dependent on the divisibility of n. The magnitude of *sect*(16) is about 10^7 and then the number straws in a haystack of the 16×16 grid must be of magnitude over 10^{12}. Thus the best two straws or needles is then really hard to find.

8. Supplement 2: GEOM in freeware version of Survo (1 Feb 2009)

The GEOM program is available also in the Survo Editor, the freeware version of SURVO MM. Survo Editor can be downloaded from
`http://www.survo.fi/english/download`

In Survo Editor the maximum number of observations is strongly limited for data files with more than 3 variables. However, this restriction is not harmful, for example, when making calculations related to approximate circle squaring and thus all tasks presented in Section 4 can be performed also in Survo Editor. For readers who like to test their own constructions, another template `_GEOM2` is available for calculations needed in approximate circle squaring. Usage of `_GEOM2` is similar to that of `_GEOM` presented in Section 6.

9. Supplement 3: Richmond's construction of regular pentagon (1 Feb 2009)

The statistical accuracy of Richmond's construction 3.5 of a regular pentagon can be improved by constructing the points A_2 and A_3 in the same way as it was done in Ptolemy's construction.

```
____________________________________________________________
*GEOM CUR+1,E,E+1
*Pentagon (Richmond 2)
*O=point(0,0)
*B=point(-1,0)
*_C1=circle(O,1)
*LY0=line(O,_B)
*LX0=perpendicular(LY0,*O)
*A0=cross_cl(C1,LX0,0,1)
*
*symbol start
*M=midpoint(*O,B,LY0)
*LAM=line(A0,M)
*LMQ=bisect_a(LAM,LY0,0,0.5)
*Q=cross(LMQ,LX0)
*LQ=perpendicular(LX0,Q)
*A1=cross_cl(C1,LQ,-1,0.2)
*A4=cross_cl(C1,LQ,1,0.2)
*E=edge(A0,A1)
*C2=circle(A1,E,1.5,1.7)
*C3=circle(A4,*E,1.3,1.5)
*A2=cross_cc(C1,C2,-1,-1)
*A3=cross_cc(C1,C3,0.5,-1)
*
*E1=edge(A1,A2)
*E2=edge(A2,A3)
*E3=edge(A3,A4)
*E4=edge(A4,A0)
*save edge(E)
*save point(A0)
*save point(A1)
```

```
*save point(A2)
*save point(A3)
*save point(A4)
E
*GEOM construction: Pentagon (Richmond 2)
*Geometrographic symbol = 8S1+4S2+12C1+0C2+9C3
*Simplicity = 33  Exactitude = 20
*
-------------------------------------------------------------
```

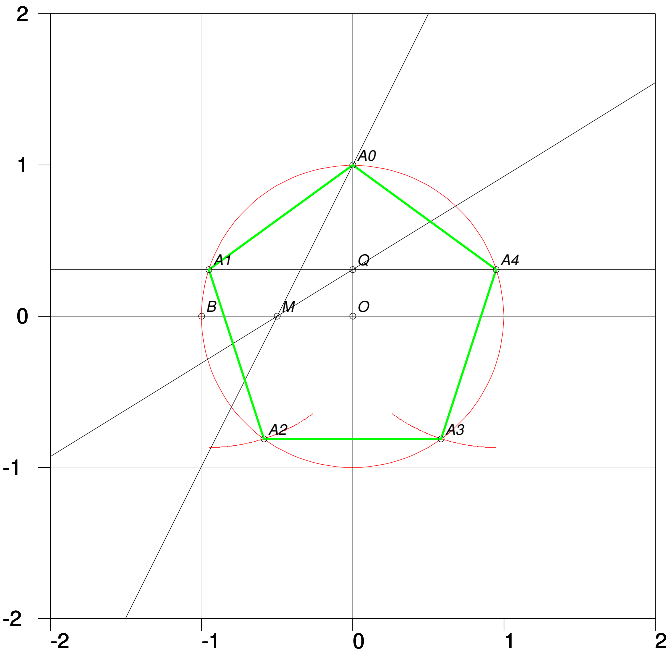


FIGURE 20. Construction of Richmond more accurately

The statistical summary of this construction is

```
-------------------------------------------------------------
*
*MINSTAT K,CUR+1  / VARS=E,D2
*Basic statistics of data K N=1000000
*Variable   mean         stddev             N  minimum      maximum
* E         1.175572666  0.002186951  1000000  1.165301756  1.186473764
* D2        0.000053988  0.000065399  1000000  0.000000021  0.001100641
*
*Accuracy acc0:
*sqrt(MAT_STAT.M(D2,mean)/5)=0.00328597788185
*
-------------------------------------------------------------
```

Thus the acc_0 value 0.003286 is clearly smaller than 0.004782 obtained in the earlier construction 3.5.

References

[1] L.Berggren, J.Borwein, P.Borwein, *Pi: A Source Book*, Springer, New York, 1997.

[2] D.W.DeTemple, Carlyle circles and the Lemoine simplicity of polygon constructions, *Amer. Math. Monthly* 98 (1991), 97–108.

[3] H.Eves, *An Introduction to the History of Mathematics*, fourth edition, Holt, Rinehart, and Winston, New York, 1976.

[4] H.R.P.Ferguson, D.H.Bailey, S.Arno, Analysis of PSLQ, an integer relation finding algorithm. *Math. Comp.* 68 (1999), 351–369.

[5] Geometry software.
`http://mathforum.org/geometry/geometry.software.html`

[6] A.L.Hess, Certain topics related to constructions with straightedge and compasses. *Math. Mag.* 29 (1956), 217–221.

[7] Hirano's construction for a pentagon.
`http://www.cut-the-knot.org/pythagoras/PentagonConstruction.shtml`

[8] H.P.Hudson, *Ruler & Compasses*, Longmans Green, London, 1916.

[9] Labelle's construction for a pentagon.
`http://www.cs.mcgill.ca/~sqrt/cons/constructions.html`
Select problem "inscribe a pentagon" and "load best known solution".

[10] É.Lemoine, *Géométrographie*, Gauthier-Villars et fils, 1893.

[11] É.Lemoine, Note on Mr. George Peirce's approximate construction for pi, *Bull. Amer. Math. Soc.* 8, (1902), 137–148.
`http://projecteuclid.org/DPubS?service=UI&version=1.0&verb=Display&handle=euclid.bams/1183416850`

[12] "Mascheroni" construction for a pentagon.
`http://www.geocities.com/robinhuiscool/Pentagon.html`

[13] T.J.Osler, A simple geometric method of estimating the error in using Vieta's product for π, *Internat. J. Math. Educ. Sci. Tech.*, 38, (2007), 136–142.
`http://www.rowan.edu/open/depts/math/osler/81%20Error%20in%20Vietas%20Product%20%20iJMEST.pdf`

[14] Ptolemy's construction for a pentagon.
`http://www.cut-the-knot.org/pythagoras/pentagon.shtml`

[15] S.Ramanujan, Modular equations and approximations of π, *Quart. J. Math.*, XLV (1914), 350–372.

[16] Richmond's construction for a pentagon.
`http://www.cut-the-knot.org/pythagoras/RichmondPentagon.shtml`

[17] H.C.Schleper, The chronology of Pi, *Math. Mag.* 23 (1950), 165–170, 216–228, 279–283.

[18] `http://www.survo.fi/english`

[19] E.W.Weisstein, "PSLQ Algorithm." From MathWorld–A Wolfram Web Resource.
`http://mathworld.wolfram.com/PSLQAlgorithm.html`

[20] W.Wieslaw, Geometry in Poland in XV–XVIII Centuries.
`http://193.0.96.201/wolne/zjazd/info/geom.ps`

The current version of this paper can be downloded from
`http://www.survo.fi/papers/GeomAccuracy.pdf`

Department of Mathematics and Statistics, University of Helsinki
E-mail address: `seppo.mustonen@helsinki.fi`