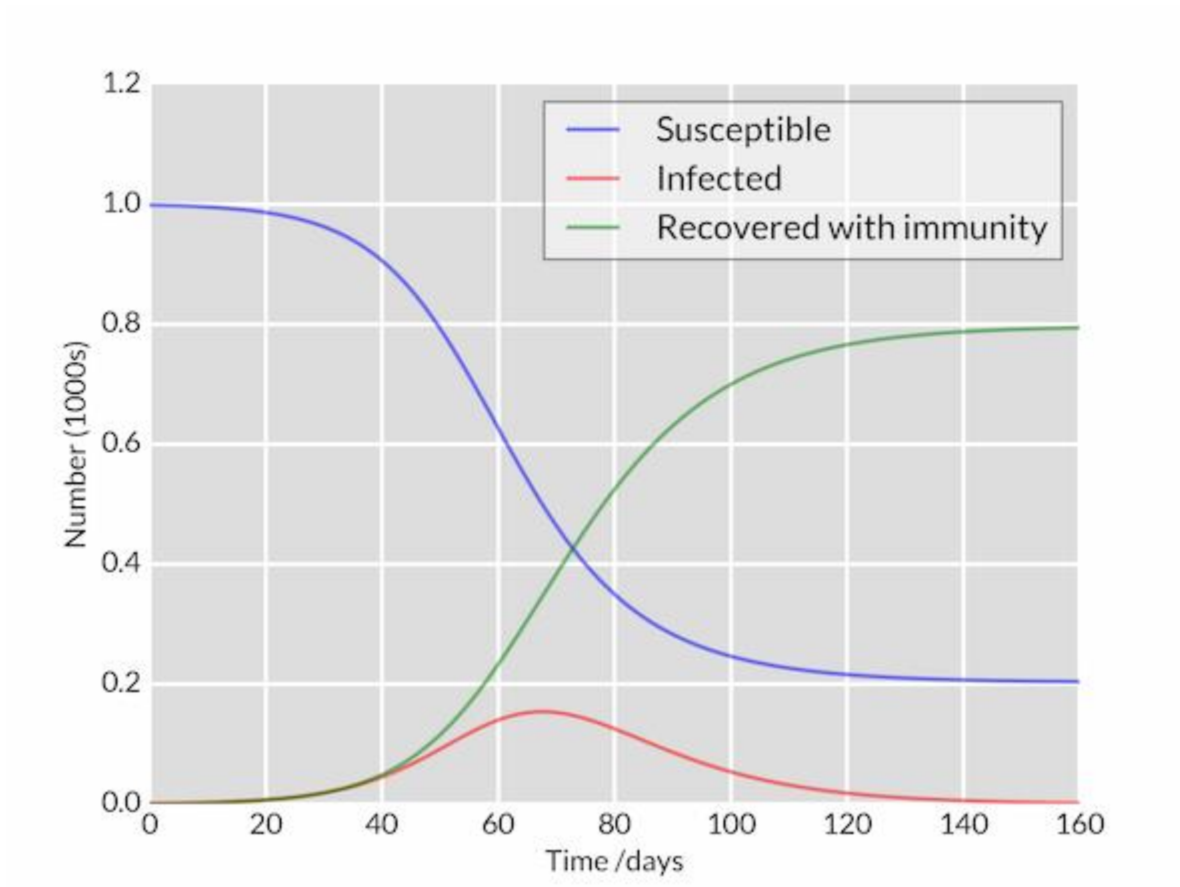
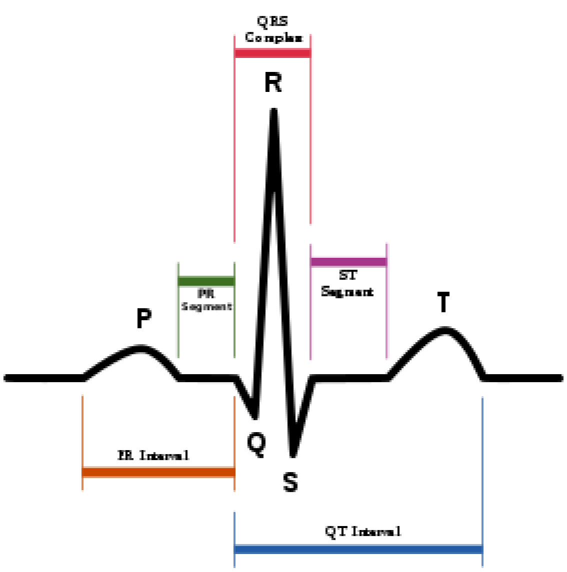
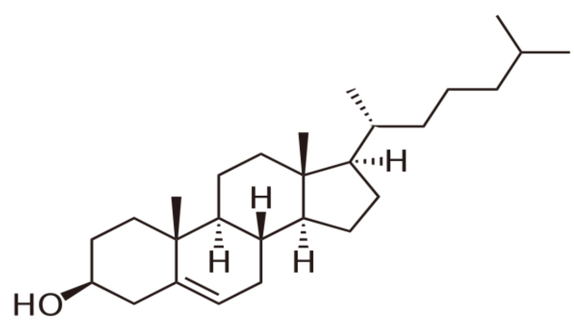
EPIDEMICS ROM

HP-41 Module







Prepared by Ángel M. Martin

September 2020

This compilation revision 1.2.2

Screen captures taken from V41, Windows-based emulator developed by Warren Furlow.
See www.hp41.org

Epidemics Module - QRG

Virus Epidemics SIR Models

Ángel Martin - JM Baillard

"SIR	Driver program	FOCAL
"d/dT	https://www.hpmuseum.org/forum/thread-14813.html	FOCAL
"*?	Auxiliary prompt	FOCAL
"RK4C	Runge-Kutta Method	FOCAL
"SIR+	Driver for "sir2	FOCAL
"SIR2	SIR model - normalized population	FOCAL
I0T	Initial Approximation	MCODE

Direct Determination of Recovered Rate

Ángel Martin

"R-RT	R=R(t) - Direct Determination - uses PPCSV		FOCAL
"R<T>	R=R(t) - Direct Determination using SIROM		FOCAL
S<R>	Susceptibles Rate		MCODE
"*RT	Function to Solve		FOCAL
"*ITX	Integrand Function		FOCAL
ITX	Aux for "*ITX		MCODE
"ITG	Numeric Integration	Internal use	
"PPCSV	PPC's Solve	Internal use	

Epidemiologic Analysis with a Programmable Calculator

E. Franco & R. Simmons

"REED	Deterministic Version	FOCAL
"FROST	Stochastic version	FOCAL
	https://www.hpmuseum.org/forum/thread-12091.html	

Intra-Ocular Lens Power Calculator

Ángel Martin

"K	Corneal Power – simple	FOCAL
"K12	Corneal Power - Double	FOCAL
"SRK/T	SRK/T IOL Calculator	FOCAL
"HAIGIS"	Haigis IOL Calculator	FOCAL

Basic EKG Calculations

Eddie W. Shore

"EKG	Driver Program	FOCAL
	https://www.hpmuseum.org/forum/thread-14075.html	

Friedewald Formula for LDL-cholesterol

Dinamarco-Dieter

"DLD-CHL	Driver Program	FOCAL
	https://www.hpmuseum.org/forum/thread-12546.html	

Correlación Ortogonal

Juan Manual Cueva-Lovelle

"CO-ORT	Driver Program	FOCAL
	http://www.hp41.org/LibView.cfm?Command=View&ItemID=1451	

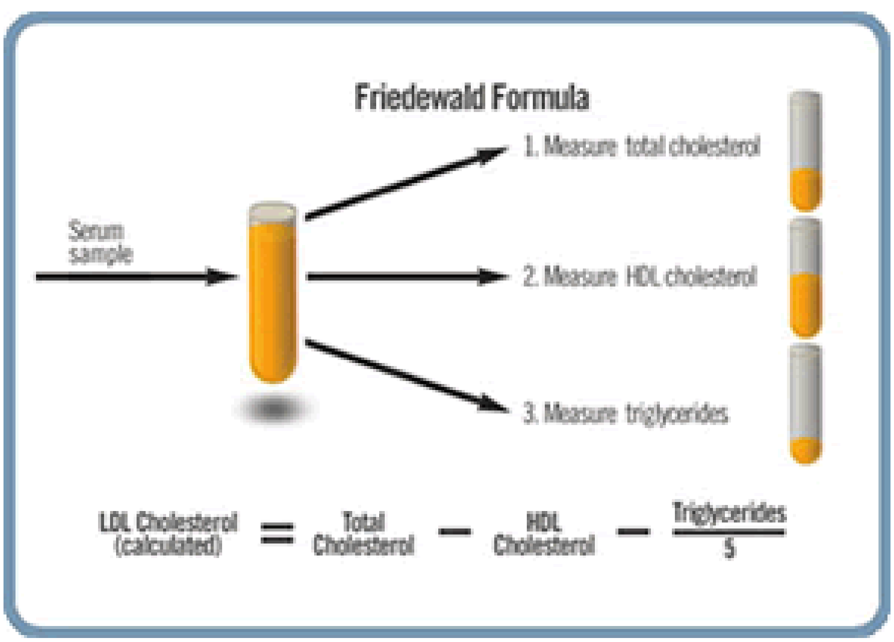
Curve Length between [a, b]		Needs SandMath	Ángel Martin
"CLEN	Curve Length w/ SandMath		FOCAL
"*CL	Integrand Function		FOCAL
"PPCIT	PPC's IT		
"CRVL	Curve Length w/ PPC Routines		FOCAL
*CV	Integrand Function		FOCAL
"PPC1D	PPC's First Derivative	Internal use	FOCAL

Cassette Tape Design			Ángel Martin
"C-TAPE	*Constant Head Speed*		FOCAL
	ETSII-4A		

Audio Tape Counter / Time Conversions (2418C)			David Hayden
"TAPE	Tape Counter		FOCAL
"TAPINT	*https://www.hpmuseum.org/forum/thread-14602.html?highlight=tape+counter*		
	UPL #2418C		

Diverse Utilities			Ángel Martin
AINT	ALPHA Integer Part		MCODE
BINET	Binet's Formula	- n in X	MCODE
PHI	Golden Ratio constant		MCODE
SIGMD	Sigmoid Function		MCODE
CTRST	LCD Contrast (Half-Nut models)		MCODE
"WOOO	Contrast Demo		FOCAL
/+/	Inverses Addition: 1/Y + 1/X		MCODE
3PMT _ _ _	Three Prompts		MCODE

Easter date finder			Kari Pasanen
EASTER	Year in X		MCODE



Epidemics ROM - Introduction.

As you can see by looking at the index, this module has an eclectic collection of programs on very unrelated subjects. The main section deals with virus epidemics, including two contributions written in collaboration with Jean-Marc Baillard, as well as an old paper on the same subject from yore.

I wrote the Intra-Ocular Lens Power calculator as a response to my frustration with the medical institutions, in particular with the lack of rigor and understanding of so-called ophthalmologists. I sincerely hope you don't know what I mean…

In the spirit of free sharing other contributions are shamelessly taken from the MoHP Forum, like the EKG Calculations by Eddie Shore and the LDL-Cholesterol by Dinamarco-Dieter – somehow still related to the medical science field.

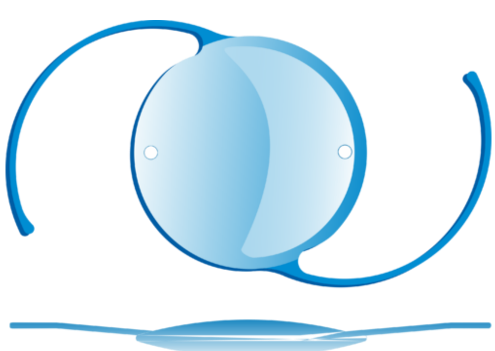
Moving onto totally unrelated subjects, the Tape Counter program from David Hayden finally motivated me to document an old mini work on cassette tape design from engineering school days, you never know when these things will come back in fashion again ;-)

A couple of other programs complete the collection, one about Orthogonal Fitting (you can never have too many of those, it seems…) plus two brute-force applications to calculate the arc length of a curve using quick & dirty routines that rely on general-purpose functions.

And last and possibly also least, a few MCODE functions either to support the FOCAL code or to fill-up the available space in the module. These include:

- Easter data finder (written by Kari Pasanen),
- display contrast Demo divertimento,
- Golden Ratio constant and Binet formula for Fibonacci numbers,
- Sigmoid function
- Sum of Inverses.

Enjoy!



Virus Epidemics - Introduction.

Chances are you're reading this from your "shelter in place" Covid-19 confinement, and thus will resonate with the concepts involved. This short program is a direct application of Jean-Marc Baillard's routines to solve a system of three ordinary differential equations (ODE).

The SIR model calculates the values of Susceptible, Infected and Removed (recovered plus dead) groups of population in a total population of N individuals. N = S + I + R.

The virus is modeled by two parameters that indicate the *infection rate* "a" and the *recovery rate* "b". These are the crux of the model, as they need to be expressed in the same units used by the ODEs to show the individual results - i.e. a=2.3 infected people per person and DAY, and b = 0.3 people recovered per DAY if we want to look at daily numbers.

Supposedly your local Covid-19 statistics could be used to estimate the parameters, but this is tricky since the reported infection cases are much lower than the actual ones. Besides, the Removed section includes both the recovered (cured) and the dead patients.

The ODEs are normalized by the total population size N, but this is transparent to the user and it's done by the routine itself. The ODE's are implemented in the LBL "d/dt" subroutine:

$$dS/dt = -a.I(t).S(t)$$
$$dI/dt = a.S(t).I(t) - b.I(t)$$
$$dR/dt = b.I(t)$$

Incremental Method.

The first approach is based on successive application of the Runge-Kutta method to solve the system of three ODE. As usual the accuracy and efficiency of this approach relies on a sensible step size and well-defined boundary conditions, with both attributes decreasing with the successive iterations (whereby each step is based on the results of the previous one).

Let S(t), R(t), and I(t) be the Susceptible, Recovered and infected segments of the population. Let's assume no vital dynamics (births and deaths), thus the total population N is constant and given by the relationship: N = S(t) + R(t) + I(t)

The non-linear differential equations that define this model are given below:

$$\frac{dS}{dt} = -\frac{\beta IS}{N},$$

$$\frac{dI}{dt} = \frac{\beta IS}{N} - \gamma I,$$

$$\frac{dR}{dt} = \gamma I,$$

Where β and γ are the constants characterizing the dynamics of the epidemic.

Note that since the total population is constant, it follows that at any given time:

$$\frac{dS}{dt} + \frac{dI}{dt} + \frac{dR}{dt} = 0,$$

Solving the Epidemic

The driver program "**SIR**" first prompts for the population size and initial stocks of susceptible and infected population, *but not the recovered segment as it is assumed null at the beginning* (t=0). It also prompt for the dynamic constants { a, b } corresponding to β and γ in the equations above.

By default, the program uses a step-size h= 0.1 and n=10 steps per calculation, which yields the results in intervals of one unit value. You can change these manually storing your custom constants in R05 and R06 respectively. As usual the time increments are given by the product of both parameters, i.e. tn = tn-1 + n.h

The system of ODEs is solved using the "classic", fourth-order Runge-Kutta method. I have used Jean-Marc's "**RK4C**" generic routine for this purpose, with the appropriate variable definitions - somehow tricky but conceptually simple.

The program assumes variables are stored in the following data registers:

R00 = subroutine name
R01 – to, then t
R02, R03, R04– initial values So, Ro, and Io
R05 = h = step size
R06 = N = number of steps
R12 – β constant
R13 – γ constant

The subroutine "**d/dT**" uses the three variables R(t), I(t) and S(t) in the { X,Y,Z} stack registers and calculates the three derivatives {dR/dt; dI/dt, and dS/dt} in stack registers { X, Y, Z } respectively.

Register	Input	Output
T:	t	-
Z:	S(t)	dS/dt
Y:	I(t)	dI/dt
X:	R(t)	dR/dt

The system of ODEs is programmed as follows - Note that there is no need to use R(t) to calculate any of the three variations: with R in X:, S in Y: and I in Z:

```
01 LBL "d/dt"     ;
02 RDN            ; S
03 X<>Y           ; I
04 *              ; S.I
05 LASTX          ; I
06 RCL 13         ; b
07 *              ;b.I
08 X<>Y           ; I.S
09 RCL 12 ; a
10 *              ;a.I.S
11 ENTER^
12 CHS            ; -a.I.S  = dS/dt to Z:
13 X<>Y           ;a.I.S
14 RCL Z          ;b.I      = dR/dt in X:
15 ST- Y   ; (a.I.S – b.I)  = dI/dt in Y:
16 RTN
```

Example:- Calculate the SIR percentages of population for an epidemic represented by the constants a = 2.25 and b = 0.3, when the normalized initial conditions are:

so = 0.9, and io = 0.1

Solution.-The program prompts for the input values, suggesting those from previous executions - in which case you can simply press R/S to reuse them:

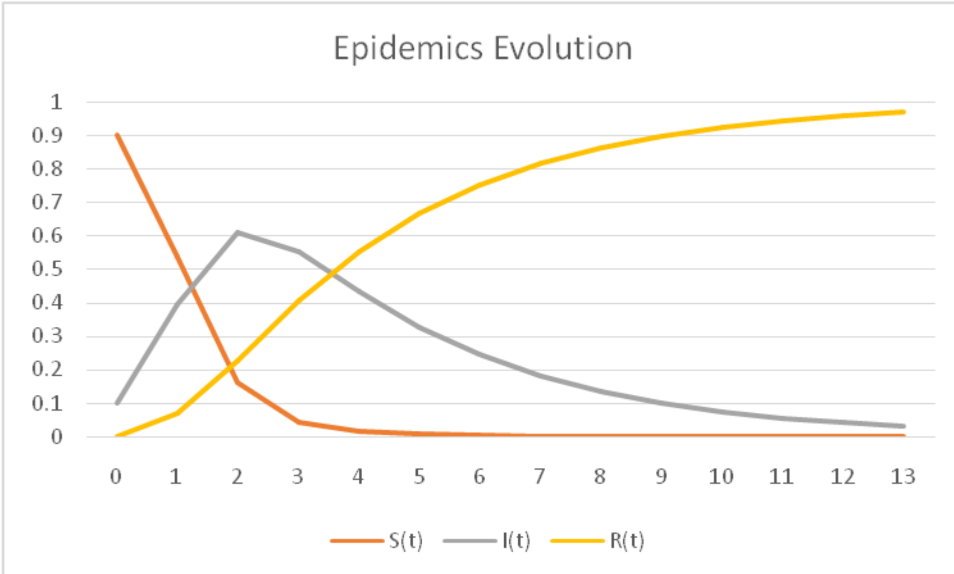
We type:	To obtain:	
XEQ "**SIR**"	a: ?	
2.25 , R/S	b: ?	
0.3 , R/S	N: ?	*total population size*
1 , R/S	S0: ?	*initial susceptible*
0.9 , RS	I0: ?	*initial infected*
0.1 , R/S		

Note that the program will check that R0=0, i.e. (S0 + I0)/N =1, and if this is not the case the prompts will be asked again until such condition is met.

Entering all the known parameters at the corresponding prompts, we obtain the following results table:

Time	S(t)	I(t)	R(t)
0	0.9	0.1	0
1	0.5366	0.3944	0.0689
2	0.1627	0.6092	0.2281
3	0.0429	0.5513	0.4058
4	0.0142	0.4324	0.5534
5	0.0061	0.3271	0.6668
6	0.0032	0.2448	0.7520
7	0.0020	0.1824	0.8157
8	0.0014	0.1356	0.8630
9	0.0011	0.1007	0.8982
10	0.0009	0.0748	0.9243
11	0.0008	0.0555	0.9437
12	0.0007	0.0412	0.9581
13	0.0006	0.0306	0.9688

Which can be represented graphically, obtaining the well-known SIR shape as expected:



Alternative program: "**SIR2+**"

This version uses a consolidated form of the different routines into a single FOCAL program, which makes it slightly faster (pre-compiled jumps instead of global label searches). The input values needed are the same as the previous case, as it is the data output sequence, the only difference being that the total population size is assumed to be one, i.e. the entry is for the **percentages** of the susceptible, infected and recovered segments.

It also uses a step size of h =0.1, and the number of steps is n =10 so the results are identical to the default case used in "**SIR**"

Here too a check is made for the initial conditions to comply with the requirement s0+ro+io=1. Should that not be the case the program will throw an error message like the one shown below:

R(0)#0

Data Register map:

R11 – a
R12 – b
R01 – to, then t
R02, R03, R04 – Initial values So, Io, Ro respectively
R05 – h/2
R06 – n

The driver program prompts for the input values and then calls "**SIR2**", where all the number crunching occurs.

Note. The table below shows the subroutines used in each of the two versions:

Routine	SIR	SIR2+	Direct Method
Runge-Kutta	RK4C	SIR2	n/a
System of OED	d/dT	LBL 10	n/a
Prompting	"*?"	"*?"	"*?"

Program Listings.-

```
01  *LBL "SIR"
02  02*LBL A
03  RCL 12
04  "a="
05  ARCL X
06  "`?"
07  PROMPT
08  STO 12
09  RCL 13
10  "b="
11  ARCL X
12  "`?"
13  PROMPT
14  STO 13
15  *LBL B
16  RCL 14
17  "N="
18  ARCL X
19  "`?"
20  PROMPT
21  STO 14
22  RCL 02
23  *
24  "S0="
25  ARCL X
26  "`?"
27  PROMPT
28  RCL 14
29  /
30  STO 02
31  RCL 03
32  RCL 14
33  *
34  "I0="
35  ARCL X
36  "`?"
37  PROMPT
38  RCL 14
39  /
40  STO 03
41  0
42  STO 04
43  STO 01
44  0.1
45  STO 05
46  10
47  STO 06
48  *LBL C
49  RCL 04
50  RCL 14
51  *
52  RCL 03
53  RCL 14
54  *
55  RCL 02
56  RCL 14
57  *
58  RCL 01
59  "S"
60  AINT
61  >"="
62  ARCL Y
63  PROMPT
64  "I"
65  AINT
66  >`="
67  ARCL Z
68  PROMPT
69  "R"
70  AINT
71  >"="
72  ARCL T
73  PROMPT
74  "d/dT"
75  ASTO 00
76  XROM "RK4C"
77  GTO C
78  RTN
79  *LBL "d/dT"
80  RDN
81  X<>Y
82  *
83  LASTX
84  RCL 13
85  *
86  X<>Y
87  RCL 12
88  *
89  ENTER^
90  CHS
91  X<>Y
92  RCL Z
93  ST- Y
94  END
```

Program Listing – SIR2+

```
*LBL "SIR2+"
*LBL A
11
"a"
XROM "?"
12
"b"
XROM "?"
*LBL B
2
"S0"
XROM "?"
3
"I0"
XROM "?"
RCL 02
+
E
X=Y?
GTO 01
TONE 0
"R0#0"
PROMPT
GTO B
*LBL 01
CLX
STO 04
STO 01
.05
STO 05
10
STO 06
*LBL C
RCL 02
RCL 03
RCL 04
RCL 01
"S("
AINT
"`)="
ARCL T
PROMPT
"I("
AINT
"`)="
ARCL Z
PROMPT
"R("
AINT
"`)="
ARCL Y
PROMPT
XROM "SIR2"
GTO C
55*LBL "SIR2"
RCL 06
STO 10
GTO 01
*LBL 10
RCL Y
*
RCL 11
*
ENTER^
CHS
X<> Z
RCL 12
*
ST- Y
RTN
*LBL 01
RCL 03
RCL 02
XEQ 10
RCL 05
ST+ 01
ST* Z
ST* T
*
STO 09
X<>Y
STO 08
RCL 03
+
R^
STO 07
RCL 02
+
XEQ 10
RCL 05
ST* Z
ST* T
*
ST+ 09
ST+ 09
X<>Y
ST+ 08
ST+ 08
RCL 03
+
R^
ST+ 07
ST+ 07
RCL 02
+
XEQ 10
RCL 05
ST+ 01
ST+ X
ST* Z
ST* T
*
ST+ 09
X<>Y
ST+ 08
RCL 03
+
R^
ST+ 07
RCL 02
+
XEQ 10
RCL 05
ST* Z
ST* T
*
RCL 09
+
3
/
ST+ 04
X<>Y
RCL 08
+
3
/
ST+ 03
R^
RCL 07
+
3
/
ST+ 02
DSE 10
GTO 01
RCL 04
RCL 03
RCL 02
RCL 01
END
```

Direct Method.

Because the third differential equation is on a single variable only, it is possible to prepare an explicit form for the equations, although requiring an iterative approach.

The resulting expression is as follows:

$$b.t = \int_0^{r(t)} \frac{dx}{io - x - so.[exp(-x\,a/b\;) - 1]}$$

Where all segments are normalized, i.e. percentual of the total population:

s = S/N;   i = I/N,   r = R/N

Here the iteration process will consist of finding the appropriate upper integration limit r(t) that makes the integral value equal to the left side of the equation, b.t – a direct function of the chosen time value. This is in turn solved as a root for the function f(t) = b.t – INTG, so you see we'll need a nested implementation of INTEG within a SOLVE loop, which explains the hefty time requirements of this implementation.

While it's true that the execution time is longer than using the successive approximation method, on the other hand it can be used to directly obtain the results at the desired time without the need to calculate all previous values.

Choosing a good initial guess is also important to reduce thenumber of iterations needed for convergence, and therefore also the total execution time. I have used the estimation for r(t) given by the formula:

$$r = -\frac{b}{a}.[(a.so - b).t - \text{Ln}(so)]$$

Once obtained the percentage of recovered population it is trivial to calculate the other two segments:

$$s(t) = so\,.\exp\left[-r(t).\frac{a}{b}\right], \qquad \text{Eq. II}$$

and from the defining relationship:

$$i(t) = 1 - r(t) - s(t)\,. \qquad \text{Eq. III}$$

and thus the population segments are fully characterized for the instant t - in a process that will need to be repeated for all instants to complete the dynamic response of the epidemic.

The module includes built-in routines to solve for the equation root ("**PPCSLV**") and to perform the numerical integral ("**ITG**"), so there is no need to use additional modules. However, the user may want to use the MCODE versions of the same routines (**FINTG** and **FROOT**) available in the SIROM Module, which for some cases will yield more accurate results in shorter time.

The main programs available are shown in the table below, as well as the subroutines used for each one:

Routine	Solve	Integration
R=RT	PPCSLV	ITG
R<T>	FROOT	FINTG

Remember that the accuracy of the solution is determined by the number of decimal places set in the calculator (FIX nn) – and that this will have a direct impact on the execution time.

Example.- Calculate the normalized population segments at the instant **t=5** for the same epidemic of the previous example, a=2.25; b=0.3; s0=0.9 and i0=0.1

Let's use FIX 4 for accuracy settings.

We type:	To obtain:
FIX 4	
XEQ "R=RT"	a=?
2.25, R/S	b=?
0.3, R/S	S0=?
0.9, R/S	I0=?
0.1, R/S	T=?
5, R/S	...*long calculation time…*
	R(T)=0.6634
R/S	S(T)=0.0062
R/S	I(T)=0.3303

- Pressing R/S here or the local label [C] at any time will prompt for a new time instant.
- Pressing the local label [A] will prompt for initial boundary conditions So, Ro, Io – then continue with time instant prompt.

R/S	T=?
1, R/S	...*long calculation time…*
	R(T)=0.0689
R/S	S(T)=0.5366
R/S	I(T)=0.3944

As you can see the results are comparable to those obtained using the successive approximation method(s) seen in previous sections.

Using the SIROM MCODE routines for the same case:

We type:	To obtain:
XEQ "R<T>"	a=?
2.25, R/S	b=?
0.3, R/S	S0=?
0.9, R/S	I0=?
0.1, R/S	T=?
5, R/S	*…long calculation time…*
	R(T)=0.6568
R/S	S(T)=0.0061
R/S	I(T)=0.3271
R/S	T=?
1, R/S	*…long calculation time…*
	R(T)=0.0689
R/S	S(T)=0.5366
R/S	I(T)=0.3944

Program Listing. - "R<T>". – Uses {R10 – R15 }

01	***LBL "ITG"**	21	XEQ IND 10	41	RCL 14
02	ASTO 10	22	STO 15	42	ST+ 11
03	STO 13	23	RCL 12	43	RCL 11
04	RDN	24	XEQ IND 10	44	XEQ IND 10
05	STO 12	25	ST+ 15	45	4
06	X<>Y	26	RCL 14	46	*
07	STO 11	27	ST+ 11	47	ST+ 15
08	08*LBL 09	28	RCL 11	48	DSE 13
09	RCL 13	29	XEQ IND 10	49	GTO 02
10	ST+ X	30	4	50	RCL 15
11	STO 13	31	*	51	RCL 14
12	2	32	ST+ 15	52	*
13	+	33	33*LBL 02	53	3
14	RCL 12	34	RCL 14	54	/
15	RCL 11	35	ST+ 11	55	RTN
16	-	36	RCL 11	56	STO 13
17	X<>Y	37	XEQ IND 10	57	GTO 09
18	/	38	ST+ X	58	END
19	STO 14	39	ST+ 15		
20	RCL 11	40	DSE 13		

Register Map.-

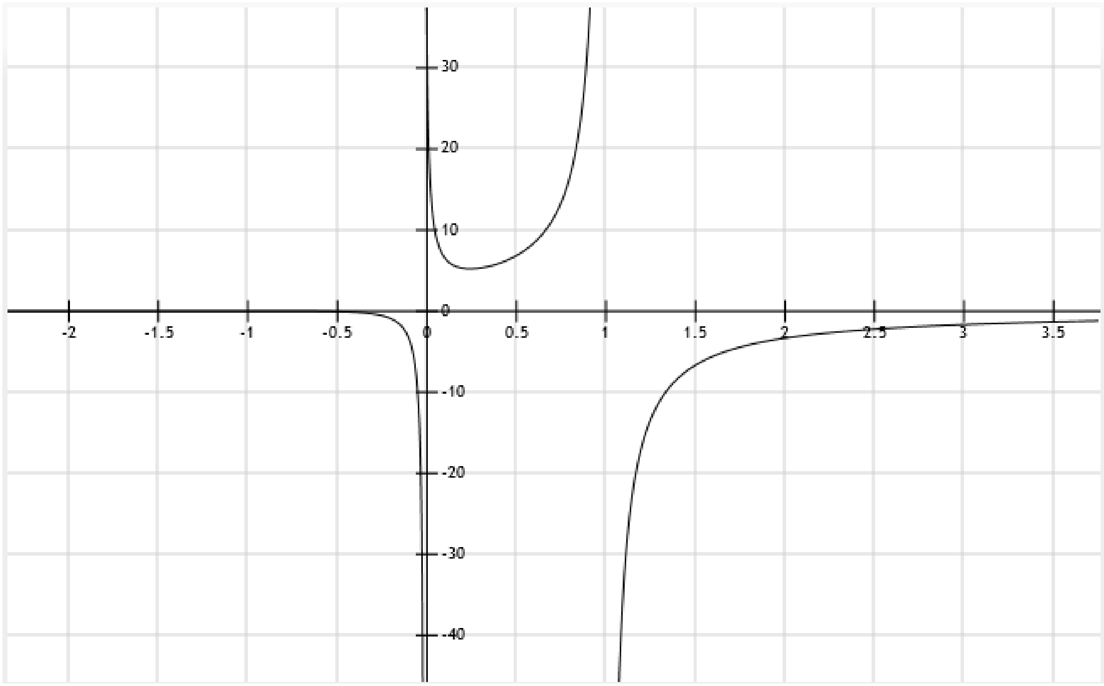
R00 – T
R01 – so
R02 – io
R03 – r(t)
R04 – a
R05 - b

```
01 *LBL "R<T>"
02 SF 01
03 GTO A
04 *LBL "R=RT"
05 CF 01
06 *LBL A
07 4
08 "a"
09 XROM "?"
10 5
11 "b"
12 XROM "?"
13 *LBL B
14 E
15 "S0"
16 XROM "?"
17 2
18 "I0"
19 XROM "?"
20 RCL 01
21 +
22 E
23 X=Y?
24 GTO 01
25 "R0#0"
26 PROMPT
27 GTO B
28 *LBL 01
29 CLX
30 STO 03
31 *LBL C
32 "T=?"
33 PROMPT
34 STO 00
35 0
36 ENTER^
37 .9
38 "*RT"
39 FC? 01
40 XROM "PPCSV"
41 FS? 01
42 FROOT
43 STO 03
44 "R(T)="
45 ARCL X
46 PROMPT
47 RCL 04
48 RCL 05
49 RCL 01
50 R^
51 S<R>
52 "S(T)="
53 ARCL X
54 PROMPT
55 RCL 03
56 +
57 E
58 -
59 CHS
60 "I(T)="
61 ARCL X
62 PROMPT
63 GTO C
64 *LBL "*RT"
65 0
66 X<>Y
67 "*ITX"
68 FS? 01
69 GTO 01
70 E1
71 XROM "ITG"
72 *LBL 01
73 FS? 01
74 FINTG
75 RCL 00
76 -
77 RTN
78 *LBL "*ITX"
79 SIGN
80 RCL 04
81 RCL 05
82 RCL 01
83 RCL 02
84 ITX
85 END
```

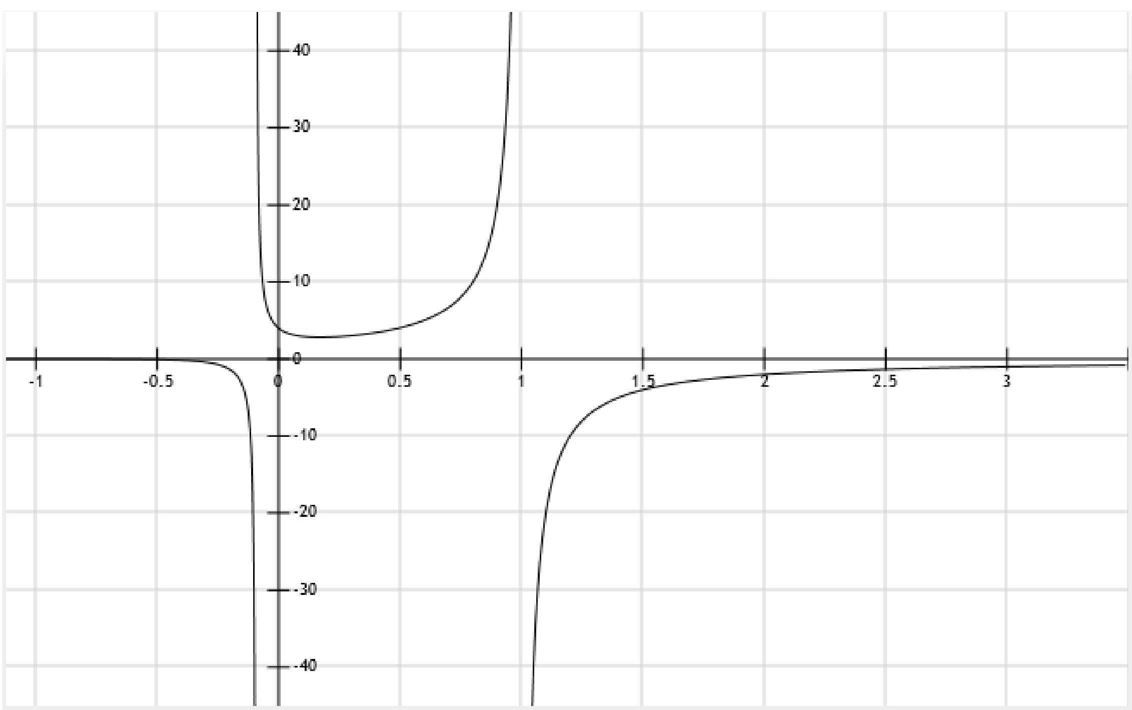
Note the call to **ITX** in step 84. This is an auxiliary MCODE function to calculate the integrand of the numeric integral, done to expedite the calculations as much as possible. The function expects the input parameters a, b, So and io in the stack registers {TZYX} respectively, and returns the value of the integrand.

Examples of integrand function, helpful to gauge the job of the numerical integral.

a= 2.5 So = 0.9 **f(x) = 1/(0.3*(0.1- x - 0.9*(exp(-2.5*x/0.3)-1)))**
b = 0.3 I0 - 0.1



a= 2.5 So = 0.5 **f(x) = 1/(0.5*(0.5- x - 0.9*(exp(-2.5*x/0.3)-1)))**
b = 0.3 I0 = 0.5



MCODE Routine listing:

!3-digit routines are used where possible, contributing to more accurate results.

Data input: *T: a* *Output: X: result*

Z: b

Y: so

$$f = \frac{1}{io - x - so.[\exp(-x\ a/b\ ) - 1]}$$

X: io

098	"X"	Integrand function
014	"T"	1 / { b.[Io - x - So.(exp(-a.x/b)-1)]
009	"I"	Ángel Martin
138	READ 4(L)	x
070	N=C ALL	
10E	A=C ALL	
046	C=0 S&X	
270	RAMSLCT	
038	READATA	a
351	?NC XQ	(includes SETDEC)
050	->14D4	[CHK_NO_S1]
135	?NC XQ	a.x
060	->184D	[MP2_10]
078	READ 1(Z)	b
2BE	C=-C-1 MS	Sign change
269	?C XQ	-a.x/b
061	->189A	[DV1_10]
048	SETF 4	substract one: e^x-1
035	?NC XQ	13-digit precision here
068	->1A0D	[EXP13]
0B8	READ 2(Y)	So
13D	?NC XQ	So.[exp(-a.x/b)-1]
060	->184F	[MP1_10]
0B0	C=N ALL	x
025	?NC XQ	x + So.[exp(-a.x/b)-1]
060	->1809	[AD1_10]
2BE	C=-C-1 MS	Sign change
11E	A=C MS	
0F8	READ 3(X)	Io
025	?NC XQ	Io - x - So.[exp(-a.x/b)-1]
060	->1809	[AD1_10]
078	READ 1(Z)	b
13D	?NC XQ	b.So.[exp(-a.x/b)-1]
060	->184F	[MP1_10]
239	?NC XQ	
060	->188E	[ON/X13
331	?NC GO	Overflow, DropST, FillXL & Exit
002	->00CC	[NFRX]

Intra-Ocular Lens Power calculator.

These routines are an approximation to IOL power calculation – a very wide and deep subject not entirely free from its dose of alchemy judging from the statements of some so-call ophthalmologists. In truth many different formulas exist, and most of them have undergone their own evolution so one talk about "generations" of formulas, each with more accuracy than the preceding one – at least in theory.

The formula used is the SRK/T – first developed by Sanders, Retzlaff and Kraff and still useful for understanding the relation of the variables involved and the IOL power.

The formula is a simple arithmetic of diopter terms, starting with a "fudge factor" known as the "A" constant - which in reality is not a constant as it depends on numerous factors like type, material and position of the IOL, as well as surgeon dependent and technique of incision.

This initial term is offset by two others, one based on the net corneal power K (keratometry value in diopters) and another on the Axial Length, AL in mm:

$$P = A - 0.9\,K - 2.5\,AL$$

This is admittedly a very crude expression, no doubt lacking refinement and other additional corrections, most notably those attempting to predict the Effective Lens Positioning (ELP) beforehand.

Corneal Power

For the calculation of the corneal power we have used a simple relationship based on the geometry of the cornea and average values of the refraction indexes in the three media: air (no), cornea (n1), and the aqueous humor inside the eye (n2) These standard values are suggested by the program. At the prompts and you can use them as default (pressing R.S) or change them as you want simply typing your own.

Default values:

n0 = 1
n1 = 1.376
n2 = 1.336

The external and internal radius are obtained during the optical biometry and are therefore known.

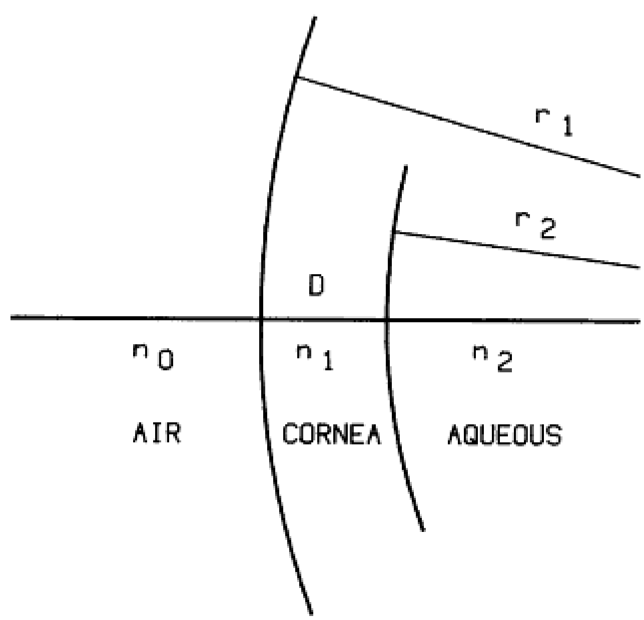
Let k1 = (n1-n0)/r1,

and K2 = (n2=n1)/r2

The expression used is as follows:

$$K = k1 + k2 - k1\,.k2.\,(r1\text{-}r2)/na$$

With na being the refractive index of the cornea and aqueous humor combined, typically of a value 1,3375.



Example.

Calculate the power of an IOL to be implanted in a patient's eye with the following biometric data:

> Axial length = 24.06 mm
> r1 = 7.97 mm
> r2 = 7.91 mm

You can assume the standard values for all refractive indexes n, na, n1 n2.

Use a "custom" A-term value: A=118.90

Solution:

We type	to obtain:
XEQ "K12"	R1 = ?
7.97 , R/S	N1 = 1,3767
R/S	R2= ?
7.91 , R/S	N2= 1.3367
R/S	K12=42.131
R/S or	
XEQ "SRK/T"	K=42.1317
R/S	AL= ?
24.06 , R/S	A= 118.1?
118.9 , R/S	P=20.832 D

The IOL to implant should have 20.832 diopters to avoid any "refractive surprises"…

Simplified Corneal Power

The module also includes another routine (LBL "**K**") to calculate the corneal power using a simplified expression, only involving the average cornea radius R and the combined refractive index na:

$$K = (na - n0)/R$$

You can use this method instead of LBL "**K12**" and plug the resulting K value into the first prompt:

Using the same example, with R = 7.94 mm

We type	to obtain:
XEQ "K"	R=?
7.94 , R/S	Na= 1.3387
R/S	K=42.506
XEQ "SRK/T"	K= ?
42.506 , R/S	AL=24.06?
R/S	A= 118.1?
118.9 , R/S	P=20.900 D

References:

https://eyewiki.aao.org/Biometry_for_Intra-Ocular_Lens_(IOL)_power_calculation

Program Listing.-

```
*LBL "K"
*LBL A
"R=?"
PROMPT
1/X
1.3375
"Na="
ARCL X
"`?"
PROMPT
E
-
*
E3
*
"K="
ARCL X
PROMPT
GTO A
RTN
*LBL "K12"
E
"R1"
XROM "?"
1.376
STO 03
3
"N1"
XROM "?"
1
-
RCL 01
/
STO 00
2
"R2"
XROM "?"
1.336
STO 04
4
"N2"
XROM "?"
RCL 03
-
RCL 02
/
RCL 00
X<>Y
ST+ 00
*
RCL 01
RCL 02
-
*
1.3375
/
ST- 00
E3
ST* 00
RCL 00
"K12="
ARCL X
PROMPT
*LBL "SRK/T"
*LBL C
0
"K"
XROM "?"
1
"AL"
XROM "?"
118.1
STO 02
2
"A"
XROM "?"
RCL 00
.9
*
-
RCL 01
2.5
*
-
STO 03
"P="
ARCL X
"` D"
PROMPT
GTO C
89 END
```

Relations between optimized IOL constants

A (SRK II)	A (SRK/T)	pACD	sf	a0	a1	a2
119.28	118.90	5.46	1.67	1.243	0.400	0.100

http://ocusoft.de/scripts2/ciolc.php?ctyp=2&cnst=118.9&subm=Convert+IOL+constant

Haigis Formula

Wolfgang Haigis introduced some corrections to the basic IOL power equation. For starters he addressed the A-constant limitations by replacing it by a system of three constants . {a0. a1, a2} and expressed the Effective Lens Position ELP as a function of them:

ELP = a0 + a1 . [ACD] + a2 . [AL]

Where [ACD] is the anterior chamber depth and [AL] is the axial length measured in the biometry.

Moreover, the constants are based on the statistical averages of axial length (23.39 mm) and anterior chamber depth (3.37 mm) across population samples, as an attempt to correlate the with experimental data, using the A-constant provider by the IOL manufacturer as starting point. This is shown below:

a0 = ACD-Constant - a1 . 3.37 a2 . 23.39

where the literature suggests the following fix values for a1 and a2:

a1 = 0.4 ; a2 = 0.1

and the following relationship links the ACD-Constant and the [A] constant provided by the IOL manufacturer (note that different sources give different values!):

ACD-Constant = 0.62467 . [A] – 68.747
ADC-Constant = 0.58357 . [A] - 63.896=

With these considerations the expression for the effective lens position results:

ELP = (0.62467.[A] – 68.747) + 0.4. [ACD] + 0.1. [AL]

Finally, the IOL power equation also considers the desired refraction Ref, added in a correction term to the corneal power value K, as shown below:

z = K + Ref/(1-Vx . Ref)] ; with:

K = corneal power, and
Vx = vertex distance = 12 mm

With these two considerations the Haigis IOL equation has the expression below:

P = n {1/(AL-ELP) – 1/(n/z – ELP]

with n = 1.336 the refraction index of the cornea.

Note: the free literature is sketchy about these terms and even has contradictory values for the averages used in the determination of a0 and other coefficients. This may be due to different population samples or to source policy, not strange in special iced scientific fields like this one that try to protect the know-how used in commercially available products.

Program Listing:

```
01*LBL "HAIGIS"
02 0
03 "K"
04 XROM "?"
05 1
06 "AL"
07 XROM "?"
08 2
09 "ACD"
10 XROM "?"
11*LBL A
12 3
13 "A"
14 XROM "?"
15 .58357
16 *
17 63.896
18 -
19 3.37
20 ENTER^
21 .4
22 *
23 -
24 23.39
25 ENTER^
26 .1
27 *
28 -
29 "a0="
30 ARCL X
31 PROMPT
32 .4
33 RCL 02
34 *
35 +
36 .1
37 RCL 01
38 *
39 +
40 STO 04
41*LBL C
42 5
43 "REF"
44 XROM "?"
45 -12 E-3
46 *
47 1
48 +
49 1/X
50 RCL 05
51 *
52 RCL 00
53 +
54 1/X
55 1.336 E3
56 *
57 RCL 04
58 -
59 1/X
60 CHS
61 RCL 01
62 RCL 04
63 -
64 1/X
65 +
66 1.336 E3
67 *
68 "P="
69 ARCL X
70 PROMPT
71 GTO C
72 END
```

Register Map:

R00 – K
R01 – AL
R02 – ADC
R03 – A
R04 – ELP
Ref – R05

Example. Using the same data as for the SRK/T section, calculate the IOL power if the ACD is 2.98 mm.

Solution:

Ref (D)	IOL Power (D)
0.5	P=19.5404
0	P=20.2756
-0.5	P=20.9986

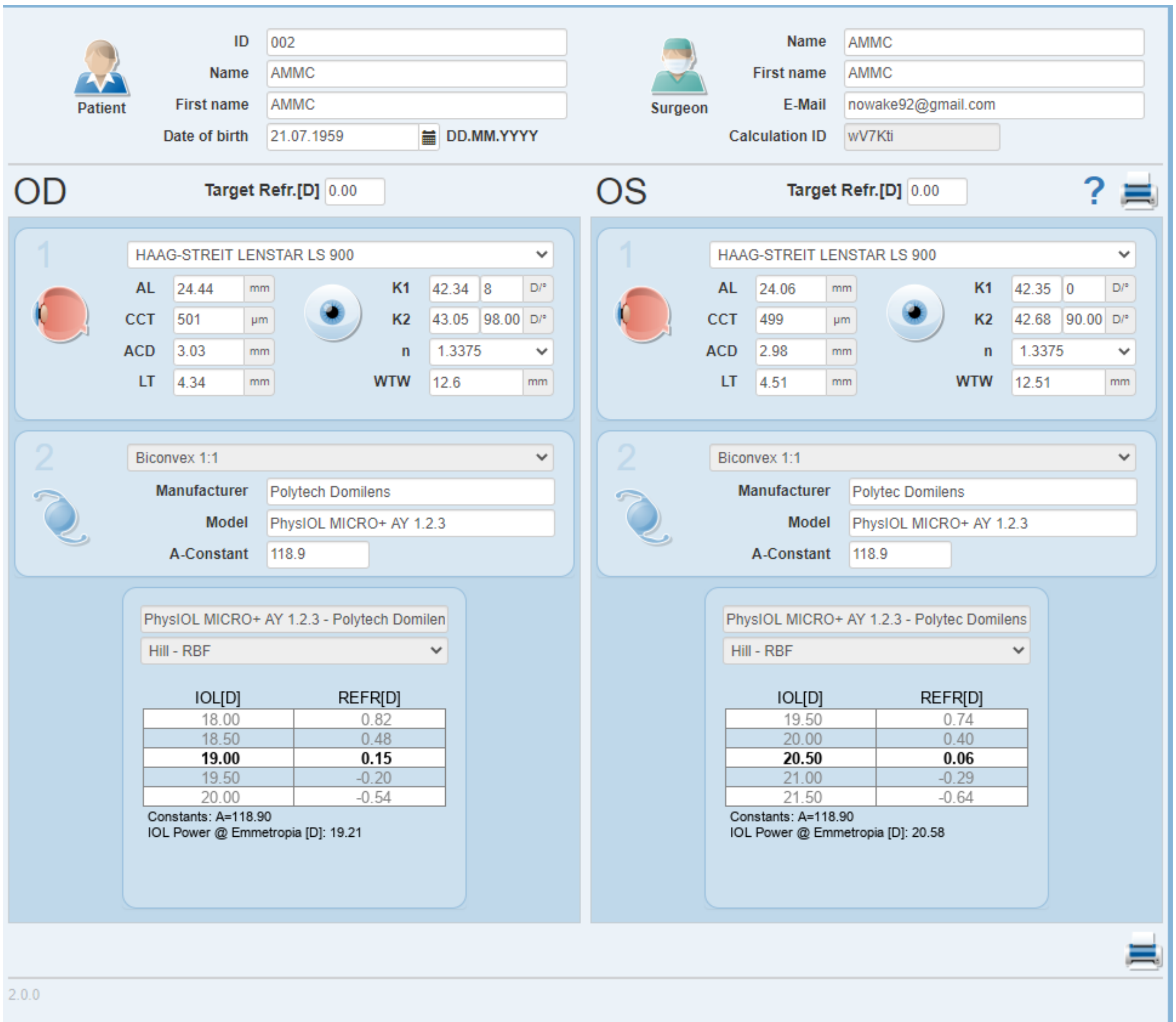
Hill-RBF: 5th.-Generation IOL Formulas

Artificial Intelligence components applied to the calculation of IOL power.

See reference paper at:
https://www.haag-streit.com/fileadmin/Haag-Streit_Diagnostics/biometry/EyeSuite_IOL/Brochures_Flyers/White_Paper_Hill-RBF_Method_20160819_2_0.pdf

See website for on-line calculator: https://rbfcalculator.com/online/index.html

The example below shows the IOL Calculation results for the author's biometry:



Curve Length - Introduction.

Two programs are included in the module with the calculation of the arc length of a curve. Both are a brute-force approach based on the definition of arc length, which requires a numerical integration of the first derivative of the curve.

$$L = \int_a^b \sqrt{1+\left(\frac{dy}{dx}\right)^2}\, dx$$

The difference between them is which routines have been used, as shown in the table below. Basically the first option is a self-contained implementation using FOCAL routines copied from the PPC ROM, whereas the second option needs the SandMath MCODE functions **DERV** and **FINTG**:

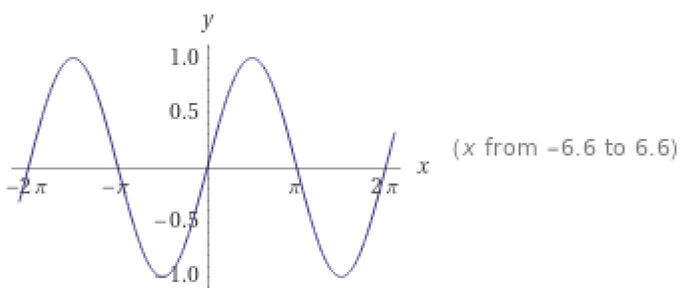
Program	Derivative	Integration
CLEN	DERV	FINTG
CRVLN	"PPC1D"	"ITG"

Be aware that both methods are ***very*** slow, therefore a TURBO emulator (or the CL) are strongly recommended. Also the decimal places determine the accuracy and the execution time.

The routines expect the abscissas delimiting the arc of the curve in the X,Y registers, and the global label name of the function in ALPHA .

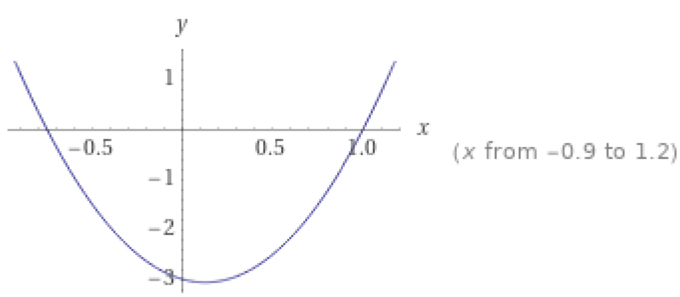
Example: Calculate the arc length between 0 and π rad of the function: y = sin (x)

Result: L = 3.820197800



Example: Obtain the arc length of $y = 4x^2-x-3$ between -0,75 and 1

Result: L = 6.517551583



The references below use more interesting approaches to the task:

https://forum.swissmicros.com/viewtopic.php?f=19&t=2302

http://hp41programs.yolasite.com/arclength.php

Program Listing.-

```
01*LBL "CRVLN"
02 ASTO 00
03 "*CV"
04 E1
05 XROM "ITG"
06 RTN
07*LBL "*CV"
08 CLA
09 ARCL 00
10 STO 08
11 .1
12 STO 02
13 8
14 STO 01
15 CF 09
16 XROM "PPC1D"
17 X^2
18 E
19 +
20 SQRT
21 CLA
22 ARCL 10
23 END
```

```
01*LBL "CLEN"            ; Needs SandMath
02 ASTO 05
03 "*CL"
04 FINTG
05 RTN
06*LBL "*CL"
07 CLA
08 ARCL 05
09 .1
10 X<>Y
11 DERV
12 X^2
13 E
14 +
15 SQRT
16 END
```

Compact Cassette Design - Introduction.

This program was part of a project on synthesis of mechanisms undertook in Engineering school many moons ago. The objective was to design a novel mechanism capable to maintain a constant sliding speed of the tape on the reading/recording head, only driven by the winding motors and thus without any friction roller on the head itself.

The program uses the equations derived from the theory to calculate the angular velocity of the driving spindle at any given time, which obviously must vary with the instant radius of the tape mini-reels to ensure a constant linear velocity at the recording head.

Geometric Dimensions

Let Ro and Rmax be the minimum and maximum radius of the tape reel, corresponding to no tape or all tape spooled in the reel respectively.

Let e be the thickness of the tape, typically expressed in mm. and L the total length, usually expressed in meters.

The general expression for the reel radius at a given time "t" is:

$$r(t) = \sqrt{(Ro^2 + \frac{e.L}{\pi})}$$

Kinematics of the problem.

Let V the linear speed of the tape sliding on the recording head, usually given in cm/s.

Let T the total time capacity of one side of the tape, i.e. T=45 min for a C-90 cassette.

The first relation between these is obviously given by: L = V.T

The formulas below give the angular velocity needed on each reel to have a constant tape linear speed V on the recording head:

$$\left\{\begin{array}{ll} \omega 1(t) = \frac{V}{r1(t)} = \frac{V}{\sqrt{(Ro^2 + \frac{e.L}{\pi})}}; & \text{for the driving spindle} \\ \omega 2(t) = \frac{V}{r2(t)} = \frac{V}{\sqrt{(Ro^2 + \frac{e}{\pi}(L - t.\ V))}}; & \text{for the rolling spindle} \end{array}\right.$$

Note that the program asks for the input values expressed in specific units, you need to convert them to them is your data is not in the same units. Likewise, the output results are given in pre-defined units as well.

Example.

What would be the outside cartridge dimension of a C-90 cassette tape with a thickness of 0.2 mm., if the spindle radius is 0.4 cm. Calculate the angular velocities of the drive and loose spindles at t=22,5 min if we want the linear tape speed on the recording head to be 2.38 cm/s

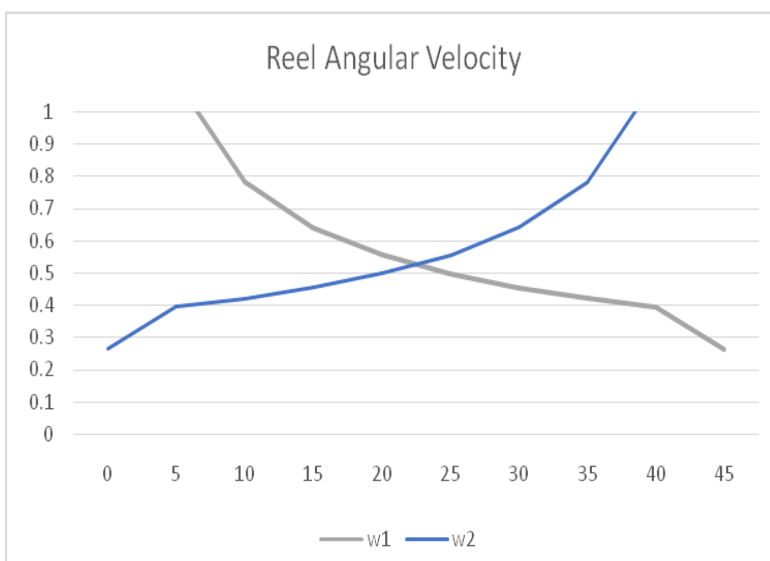
Solution.

We type:	to Obtain:
XEQ "C-TAPE"	R0=? (CM)
0.4, R/S	V=? (CM/S)
2.38, R/S	e=? (MM)
0.2, R/S	C-T=? (MIN)
45, R/S	L=64.2600 M
R/S	RMAX=0.0641 M
R/S	T=? (S)
1350 , R/S	W1=0.5242 Rad/S
R/S	W2=0.5242 Rad/S

Shouldn't surprise us that both angular velocities are equal because 45 min is the mid-point of the tape, where r1(t) = r2(t). Pressing R/S at this point takes the execution to a new time prompt, (LBL B), so you can prepare a table of angular velocities as a function of time with repeated executions.

.

t (min)	w1	w2
0	5.95	0.2629
5	1.0972	0.3938
10	0.7825	0.4209
15	0.6408	0.4544
20	0.5557	0.4975
25	0.4975	0.5557
30	0.4544	0.6408
35	0.4209	0.7825
40	0.3938	1.0972
45	0.2629	5.95



Register Map.-

R00 – Ro in cm
R01 – V in cm/s
R02 – e in mm
R03 – T in min
R04 – L = [60.T * V/100] in m
R05 – Rmax = sqrt(Ro^2 + e.L/π)

Program Listing.

```
01*LBL "C-TAPE"
02 RCL 00
03 "R0=? (CM)"
04 PROMPT
05 STO 00
06 RCL 01
07 "V=? (CM/S)"
08 PROMPT
09 STO 01
10 RCL 02
11 "e=? (MM)"
12 PROMPT
13 STO 02
14*LBL A
15 RCL 03
16 "C-T=? (MIN)"
17 PROMPT
18 STO 03
19 60
20 *
21 RCL 01
22 E2
23 /
24 *
25 STO 04          ; L
26 "L="
27 ARCL X
28 >" M"
29 PROMPT
30 RCL 00
31 E2
32 /
33 X^2
34 RCL 02
35 E3
36 /
37 RCL 04
38 *
39 PI
40 /
41 +
42 SQRT
43 STO 05          ; Rmax
44 "RMAX="
45 ARCL X
46 >" M"
47 PROMPT
48*LBL B
49 "T=? (S)"
50 PROMPT
51 STO 06
52 RCL 01          ; V
53 E2
54 /
55 *               ; V.t
56 RCL 02          ; e
57 E3
58 /
59 *               ; e.V.t
60 PI
61 /
62 RCL 00          ; Ro
63 E2
64 /
65 X^2             ; Ro^2
66 +
67 SQRT
68 1/X
69 RCL 01          ; V
70 E2
71 /
72 *               ; w1
73 "W1="
74 ARCL X
75 >" Rad/S"
76 STO 07
77 PROMPT
78 RCL 04          ; L
79 RCL 06          ; t
80 RCL 01          ; V
81 E2
82 /
83 *               ; V.t
84 -               ; (L – V.t)
85 RCL 02          ; e
86 E3
87 /
88 *               ; e.(L-V.t)
89 PI
90 /
91RCL 00
92 E2
93 /
94X^2             ; Ro^2
95 +
96 SQRT
97 1/X
98 RCL 01
99 E2
102 /
103 *
104 "W2="
105 ARCL X
106 >" Rad/S"
107 PROMPT
108 GTO B
109 END
```

Display Contrast Demo.

This small divertimento showcases the contrast setting capability in the HalfNut models (and V41 emulator). It's supposed to ne a spooky sequence of "WOOO…." displays with varying contrast settings, from faintest to darkest – repeated cyclically. Just fun fun!

…

The demo has two components:

- A contrast-setting MCODE function that uses the value in X to set the darkness of the LCD font, ranging from 0 to 15. This function is taken from the HEPAX module where it first appeared.
- A short FOCAL routine that drives **CTRST** in a repeated loop exercising all ranges, first ascending with ISG control and then descending with DSE control.

Program Listing:

01 LBL "WOOO"
02 "WOOOOOOOO, , ,"
03 AVIEW
04 LBL 02
05 ,015
06 LBL 00
07 INT
08 CTRST
09 LAST X
10 ISG X
11 GTO 00
12 15
13 LBL 01
14 INT
15 CTRST
16 LAST X
17 DSE X
18 GTO 01
19 GTO 02
20 END

094	"T"	
013	"S"	
012	"R"	
014	"T"	
003	"C"	*Michael Katz*
0F8	**READ 3(X)**	
38D	*?NC XQ*	
008	*->02E3*	*[BCDBIN]*
0A6	A<>C S&X	
130	LDI S&X	
010	CON: 16	
306	?A<C S&X	
01F	*JC +03*	
085	*?NC GO*	
0A2	*->2821*	
270	RAM SLCT	
3F0	PRPH SLCT	
0A6	A<>C S&X	
168	WRIT 5(M)	
149	*?NC GO*	*Select Chip0*
026	*->0952*	*[ENCP00]*

Other Auxiliary Functions.

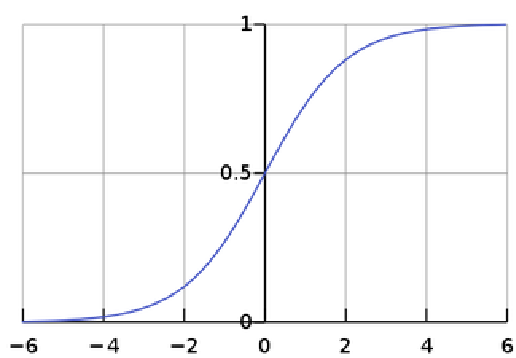
A few auxiliary functions are also included in the module to support the FOCAL programs or to complete the sections taking advantage of available space in the ROM. These are listed in the table below:

Function	Description	Notes	Author
AINT	Appends integer to ALPHA	Uses value in X register	*Fritz Ferwerda*
SIGMD	Sigmoid Function	Uses input in X:	*Ángel Martin*
/+/	Inverses Sum	1/x + 1/y	*Ángel Martin*
3PMT	Three Prompts	Prompts for input	*Ángel Martin*
EASTER	Calculates Easter Date	Year in X:	*Kari Pasanen*

Routines Listing.-

0AF	"/"	***Inverses Sum***
02B	"+"	*1/x + 1/y*
02F	"/"	*Ángel Martin*
1A5	?NC XQ	*Check for valid entries*
100	->4069	*[CHKST2]*
22D	?NC XQ	
060	->188B	*[1/X_10]*
089	?NC XQ	*1/x*
064	->1922	*[STSCR]*
0B8	READ 2(Y)	
22D	?NC XQ	
060	->188B	*[1/X_10]*
0D1	?NC XQ	*1/x*
064	->1934	*[RCSCR]*
031	?NC XQ	*1/x + 1/y*
060	->180C	*[AD2-13]*
369	?NC GO	*Overflow, DropST, FillXL & Exit*
002	->00DA	*[NFRXY]*
084	"D"	
00D	"M"	***Sigmoid Function***
007	"G"	*sig = 1/(1 + e^-x)*
009	"I"	
013	"S"	*Ángel Martin*
0F8	**READ 3(X)**	
361	?NC XQ	*(includes SETDEC)*
050	->14D8	*[CHK_NO_S]*
2BE	C=-C-1 MS	*Sign change*
044	CLRF 4	*standard version (w/out "-1")*
029	?NC XQ	*e^-x*
068	->1A0A	*[EXP10]*
001	?NC XQ	*1+e^-x*
060	->1800	*[ADDONE]*
239	?NC XQ	*1/(1+e^-x)*
060	->188E	*[ON/X13]*
331	?NC GO	*Overflow, DropST, FillXL & Exit*
002	->00CC	*[NFRX]*

$$S(x) = \frac{1}{1+e^{-x}} = \frac{e^x}{e^x+1}.$$



Friedewald Formula to evaluate LDL Cholesterol

https://blog.healthmatters.io/2018/05/28/how-to-calculate-your-ldl-cholesterol/

LDL Cholesterol is often not measured directly, but calculated using an equation which uses the other components of the lipid profile: Total Cholesterol, HDL Cholesterol and Triglycerides. If the LDL is not calculated directly, we call this "**Friedewald derived LDL-cholesterol**".

If calculated using all concentrations in **mg/dL** then the equation is:

[LDL] = [Total Cholesterol] – [HDL] – [(Triglycerides/5)]

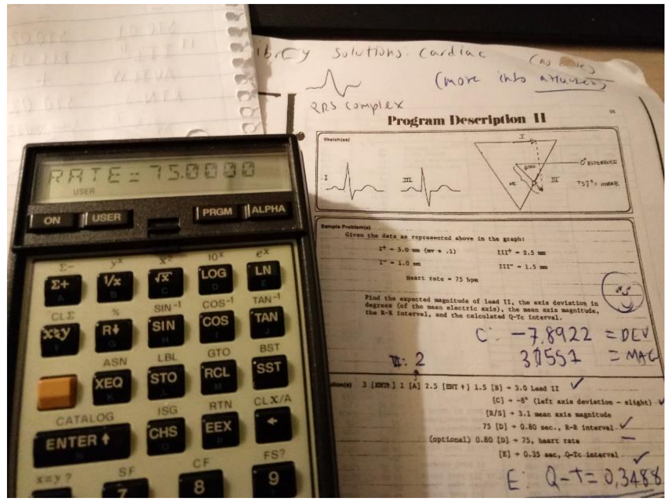
Take your total cholesterol level, subtract the HDL, and subtract the triglycerides (that have been divided by 5).

The Friedewald formula is known to be quite inaccurate at extremes of triglycerides and total cholesterol [R]. The difference between measured and calculated LDL-C increases as the triglyceride level increases [R].

Program Listing.-

```
01*LBL "LDL-CHL"
02 "TOT-C? MG/DL"
03 PROMPT
04 "HDL-C? MG/DL"
05 PROMPT
06 -
07*LBL 02
08 "TRIGL? MG/DL"
09 PROMPT
10 400
11 X>Y?
12 GTO 01
13 "MUST BY <"
14 AINT
15 RDN
16 AVIEW
17 PSE
18 GTO 02
19*LBL 01
20 RDN
21 5
22 /
23 -
24 "LDL-L="
25 AINT
26 AVIEW
27 END
```

HP 41C and DM41L: Basic EKG Calculations



Introduction

The program calculates the following:

* Lead II magnification, in mm
* The mean axis deviation
* The mean axis magnitude

Given:

* Lead I positive deflection, in mm
* Lead III negative deflection, in mm

The mean axis deviation and magnitude are calculated by a rectangular conversion by the following coordinates:

X: Lead I positive deflection - Lead I negative deflection
Y: Net Lead I deflection * 0.5774 + Net Lead III deflection * 1.1547
Mean axis deviation: θ - 57°

The program also:

* Converts between the heart rate (rpm) and the R-R interval
* Use either parameter to calculate the Q-T interval, in seconds

heart rate = 60/R-R
Q-T interval = √(R-R)*0.39

The program is a translation of HP 67/HP 97 Basic EKG Determination, which itself is a translation of Steven A. Conrad's HP-65 program (HP-65 Users' Library Program). See the source listed below.

Notes:

1. Clear the assignments of keys [**A**] (Σ+) through [**E**] (LN) before running the program. This must be done outside the programming environment. Clear assignments to keys by ASN (blank) (designated key).

Example: Clear assignment from [A]: [shift] (ASN) [ALPHA] [ALPHA] [Σ+]

2. Running EKG will turn the User Keyboard on.

3. The program will set the calculator to degrees mode.

4. This program was entered on an HP 41C, and it should work on any simulator and Swiss Micros DM41.

Instructions
Assuming USER mode is ON

1. Run EKG.

2. Determine Lead I net deviation and store it to register 01:
positive deviation Lead I, [ENTER], negative deviation Lead I , [**A**]

3. Determine Lead II net deviation, storing net Lead III net deviation to register 03 and Lead II net deviation to register 02:
positive deviation Lead III, [ENTER], negative deviation Lead III, [**B**]
Result: Lead II net deviation

4. Compute Mean Axis:
[**C**] deviation is displayed, [R/S] magnitude is displayed

5a. Convert heart rate to R-R:
heart rate (bpm), [**D**]

-or-

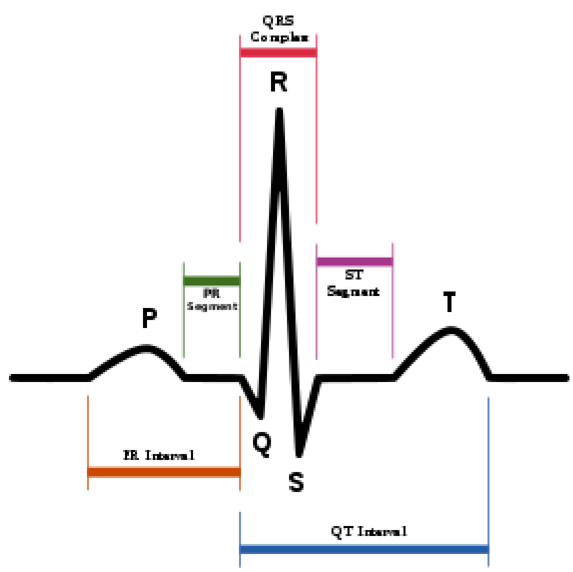
5b. Convert R-R to heart rate:
R-R, [**D**]

6. Compute Q-T.
press [**E**]

Example

I+ = 2.8 mm, I- = 1.1 mm
III+ = 2.5 mm, III- = 1.4 mm
Heart Rate = 86 bpm

Keystrokes:	Result
(User Mode is On)	
XEQ "EKG"	- -
2.8, ENTER, 1.1 [**A**]	III: 1.7000
2.5, ENTER, 1.4, [**B**]	II: 2.8000
[**C**]	DEV=-4.0516
[R/S]	MAG=2.8214
86, [**D**]	R-R=0.6977
[**E**]	Q-T=0.3258



Source:

"Basic EKG Determinations" HP-67/97 User's Library Solutions: Cardiac. Hewlett Packard. Corvallis, OR. (no date given, but I estimate this to be circa 1974)

HP 41C/DM41 Program: EKG

```
01 LBL "EKG"
02 SF 27
03 GTO 00
04 LBL A
05 -
06 STO 01
07 "III:_"
08 ARCL X
09 AVIEW
10 RTN
11 LBL B
12 -
13 STO 03
14 RCL 01
15 +
16 STO 02
17 "II:_"
18 ARCL X
19 AVIEW
20 RTN
21 LBL C
22 RCL 01
23 ENTER↑
24 0.5774
25 *
26 RCL 03
27 ENTER↑
28 1.1547
29 *
30 +
31 ENTER↑
32 RCL 01
33 R-P
34 X<>Y
35 57
36 -
37 "DEV="
38 ARCL X
39 AVIEW
40 STOP
41 X<>Y
42 ABS
43 "MAG="
44 ARCL X
45 AVIEW
46 RTN
47 LBL D
48 ENTER↑
49 ENTER↑
50 60
51 X<>Y
52 /
53 X<=Y?
54 GTO 05
55 "RATE="
56 ARCL X
57 AVIEW
58 RTN
59 LBL 05
60 "R-R="
61 ARCL X
62 AVIEW
63 RTN
64 LBL E
65 X>Y?
66 X<>Y
67 SQRT
68 0.39
69 *
70 "Q-T="
71 ARCL X
72 AVIEW
73 RTN
74 LBL 00
75 RTN
76 END
```

Appendixes.

Program Title Audio tape counter / time conversions

Contributor's Name David Hayden

Address 2 Farnsworth Ave

City Bordentown **State/Country** NJ **Zip Code** 08505

Program Description, Equations, Variables:

Tape Counters measure not how much tape has gone by the heads, but rather how many revolutions one reel has undergone. Because the amount of tape that wraps onto a reel depends on how much tape is already there, the tape counter does not directly reflect the length of tape (and hence time) that has elapsed. The actual relation between counter reading and elapsed time is:

$$t = \frac{2\pi r_0 \alpha}{v} c + \frac{T\pi\alpha^2}{v} c^2$$

Where:

t = elapsed time
α = revolutions per counter reading
c = counter reading
r_0 = radius of tape hub
v = tape velocity (speed)
T = tape thickness.

See the appendix for the derivation of this formula. T, v, r_0, π, and α are all constants so the formula is simply $t = Ac + Bc^2$. The TAPINT procedure figures these constants from easily found experimental data. Once the constants are stored, the program is fully operational. You can skip passages (given counter reading at beginning of passage and length of passage to get counter reading at end of passage), figure elapsed time from counter reading, and vice versa, and get the elapsed time between two readings.

Necessary Accessories The program works better if you have a tape deck to use it with…. Aside from that, any HP-41 can use it.

Operating Limits and Warnings Be as accurate as possible when using the TAPINT program, your results will only be as good as your measurements. You will need to re-run TAPINT should any of the variables T, v, r_0, or α change. Note that 60, 90, and 120 minute cassette tapes <u>all</u> have different thicknesses. Finally, the program assumes that counter reading 0000 corresponds to a fully rewound tape.

References This one's entirely my fault.

2418C PROGRAM DESCRIPTION II

Sample Problem: Determine constants for a 90 minute tape on a Nakamichi 480 and play with the program.

Put a 90 minute tape in the deck, rewind it, reset the counter, and start it playing. After exactly 20 minutes, the counter reads 406. After 45 minutes it reads 749.

1. A song begins at counter reading 214 and is 6m24s long. At what reading does it end?
2. Another song starts at counter reading 314 and ends at counter reading 510. How long is it?
3. A tape has been playing for 8m34s. What is the counter reading?
4. If the counter reads 522, for how long have you been listening to the tape?

INPUT	FUNCTION	DISPLAY	COMMENTS
Load TAPINT			
	XEQ "TAPINT"	TIME1?	Enter time in MM.SS format
20.00	RUN	COUNT1?	Enter corresponding counter reading.
406	RUN	TIME2?	Enter second time in MM.SS format
45.00	RUN	COUNT2?	Enter corresponding counter reading
749	RUN		Constants A and B are stored and the program is ready to run.
Load TAPE and set USER mode			
1.			
	XEQ A	COUNT=?	Enter initial counter reading
214	RUN	TIME (MM.SS)?	Enter time in MM.SS format.
6.24	RUN	COUNT=333.	The desired counter reading
2.			
	XEQ B	COUNT1?	Enter first counter reading
314	RUN	COUNT2?	Enter second counter reading
510	RUN	TIME=12.14	Time in MM.SS format
3			
	XEQ C	MM.SS?	Enter time
8.34	RUN	COUNT=200	Desired counter reading
4.			
	XEQ D	COUNT?	Enter counter reading
522	RUN	TIME=27.37	Desired time

TAPE Program

SIZE: 011
(HP-41C)

STEP	INSTRUCTIONS	INPUT	FUNCTION	DISPLAY
1	Ensure constants A and B are in proper registers. Run TAPINT if not.			
2	Enter program, set USER mode.			
3	**To skip a passage:**		(A)	COUNT=?
	Enter counter reading at beginning of passage	c	RUN	TIME(MM.SS)?
	Enter time of passage in correct format	MM.SS	RUN	COUNT=ccc
	To Calculate time between two readings:		(B)	COUNT1?
	Enter first reading	c1	RUN	COUNT2?
	Enter second reading	c2	RUN	TIME=MM.SS
	To calculate reading from elapsed time:		(C)	MM.SS?
	Enter elapsed time	MM.SS	RUN	COUNT=ccc
	To calculate elapsed time from counter reading		(D)	COUNT?
	Enter counter reading	c	RUN	TIME=MM.SS
4.	To repeat any conversion, go to step 3			

TAPINT Program

STEP	INSTRUCTIONS	INPUT	FUNCTION	DISPLAY
1	Enter program			
2	Put a tape of the desired type into/on the machine. Rewind it, reset the counter and then simultaneously start the tape playing and start a timer. A wrist watch with a sweep second hand will do, a stop watch is better. After the tape is about half way done, note the counter reading and the exact time elapsed. You will probably want to write these values down. When the tape is almost done, take the exact time for another counter reading.			
3	Calculate the constants:		XEQ “TAPINT”	TIME1?
	Enter time in MM.SS format from the first reading – elapsed time pair.	t1	RUN	COUNT1?
	Enter counter reading for the above time	c1	RUN	TIME2?
	Enter second time reading (MM.SS)	t2	RUN	COUNT2?
	Enter corresponding counter reading	c2	RUN	
	The constants A and B are now stored in their proper registers.			

STEP/ LINE	KEY ENTRY	KEYCODE (67/97 only)	COMMENTS
01	LBL "TAPE"		
*02	LBL A		
03	"COUNT=?"		
04	PROMPT		
05	XEQ 21		Count to seconds
06	"TIME <MM.SS>?"		
07	PROMPT		
08	HR		Convert MM.SS to
09	60		decimal seconds
10	*		
11	+		
12	XEQ 22		Seconds to count
13	RTN		
14	LBL B		
15	"COUNT1?"		
16	PROMPT		
17	XEQ 21		Count1 to seconds
18	"COUNT2?"		
19	PROMPT		
20	XEQ 21		Count2 to seconds
21	-		Time1 – Time2
22	ABS		Make time positive
23	60		Convert seconds
24	/		to MM.SS
25	HMS		
26	GTO 23		Display time
27	LBL C		
28	"MM.SS?"		
29	PROMPT		
30	HR		Convert MM.SS to
31	60		seconds
32	*		
33	XEQ 22		Seconds to count
34	RTN		
*35	LBL D		
36	"COUNT?"		
37	PROMPT		
38	XEQ 21		Count to seconds
39	60		Seconds to MM.SS
40	/		
41	HMS		
42	GTO 23		Display time
*43	LBL 21		count to seconds
44	RCL X		c
45	RCL 10		B
46	*		cB
47	RCL 09		A
48	+		A+cB
49	*		c(A+cB) = $Ac+Bc^2$
50	RTN		
51	LBL 22		seconds to count
52	RCL 10		B
53	/		t/B
54	RCL 09		A
55	RCL 10		B
56	/		A/B
57	2		
58	/		A/2B
59	STO Z		
60	X^2		$(A/2B)^2$
61	+		$t/B + (A/2B)^2$
62	SQRT		
63	X<>Y		
64	-		SQRT(t/B + $(A/2B)^2$) – t/B
65	FIX 00		Display count
66	"COUNT="		
67	ARCL X		
68	FIX 02		
69	AVIEW		
70	RTN		
71	LBL 23		Display time
72	FIX 02		
73	"TIME="		
74	ARCL X		
75	AVIEW		
76	END		

2418C PROGRAM LISTING

STEP/ LINE	KEY ENTRY	KEYCODE (67/97 only)	COMMENTS
01	LBL "TAPINT"		
02	"TIME1?"		Input 2 pairs of
03	PROMPT		experimental data
04	HR		
05	60		Convert to seconds
06	*		
07	STO 01		
08	"COUNT1?"		
09	PROMPT		
10	STO 02		
11	"TIME2?"		
12	PROMPT		
13	HR		
14	60		
15	*		
16	STO 03		
17	"COUNT2?"		
18	PROMPT		
19	STO 04		
20	RCL 02		Calculator
21	RCL 04		determinant
22	X^2		
23	*		
24	RCL 04		
25	RCL 02		
26	X^2		
27	*		
28	-		
29	STO 00		
30	RCL 01		
31	RCL 04		
32	X^2		Calculate and
33	*		store A
34	RCL 03		
35	RCL 02		
36	X^2		
37	*		
38	-		
39	RCL 00		
40	/		
41	STO 09		
42	RCL 02		
43	RCL 03		
44	*		Calculate and
45	RCL 04		store B
46	RCL 01		
47	*		
48	-		
49	RCL 00		
50	/		
51	STO 10		
52	END		

See the appendix for an explanation of how this works.

REGISTERS, STATUS, FLAGS, ASSIGNMENTS

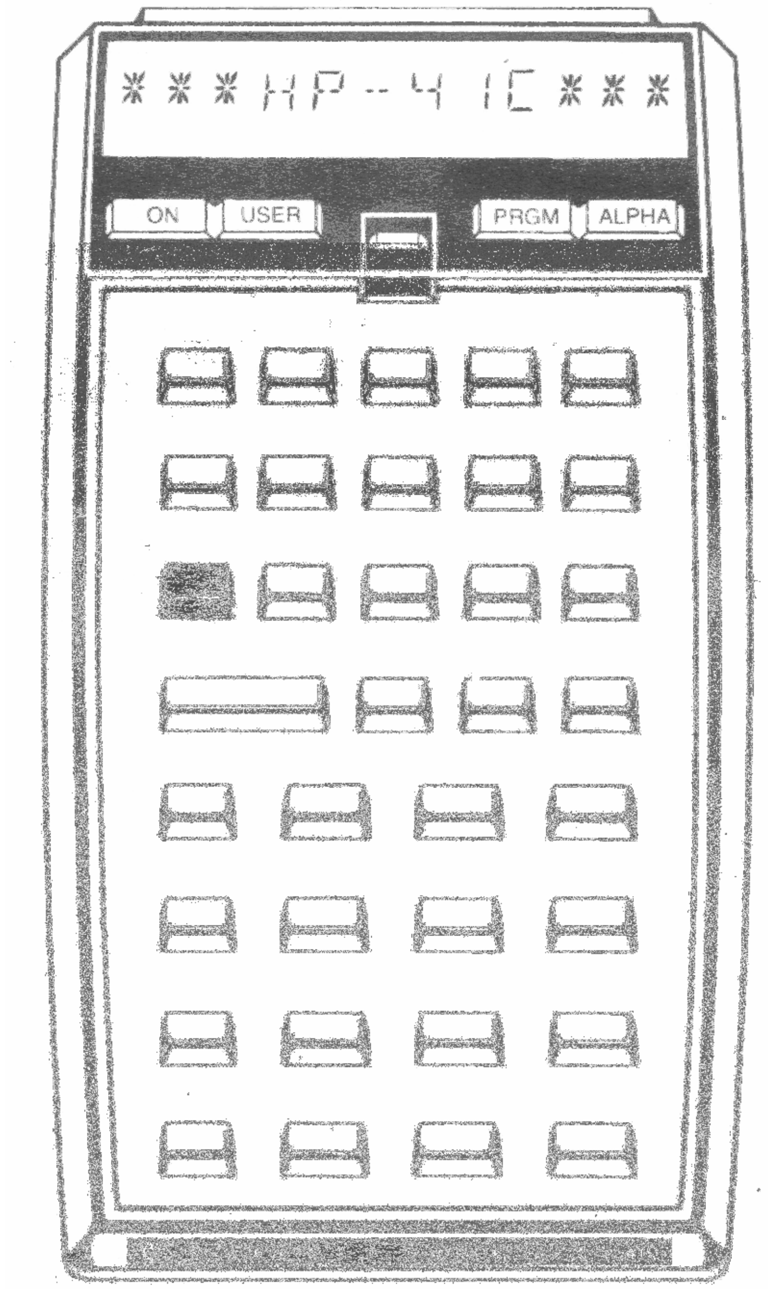
TAPE

DATA REGISTERS	STATUS			
R09 A	SIZE 011	TOT. REG. 037		USER MODE
R10 B	ENG	FIX 2	SCI	ON XX OFF
	DEG	RAD	GRAD	
	FLAGS			
	#	INIT S/C	SET INDICATES	CLEAR INDICATES
	ASSIGNMENTS			
	FUNCTION	KEY	FUNCTION	KEY

TAPINT

DATA REGISTERS		STATUS			
R00	Determinant	SIZE 011	TOT. REG. 037		USER MODE
R01	Time 1	ENG	FIX 2	SCI	ON XX OFF
R02	Count 1	DEG	RAD	GRAD	
R03	Time 2	**FLAGS**			
R04	Count 2	#	INIT S/C	SET INDICATES	CLEAR INDICATES
R09	A	**ASSIGNMENTS**			
R10	B	FUNCTION	KEY	FUNCTION	KEY

KEYBOARD



SYSTEM CONFIGURATION

CARD

		TAPE		Track 1,2 c
Count, time → count	Count, count → time	Time → count	Count → time	

Derivation of general formula

At any given time, the length of tape *s* that winds onto the take-up reel is given by the equation

$$ds = r d\theta$$

where *r* is the radius of the circle formed by the hub and the tape already on the hub, and θ is in radians. But at any time, the radius of this circle is given by

$$r = r_0 + \frac{\theta}{2\pi} T$$

where *T* is the thickness of the tape. This equation says that the radius is equal to the radius of the hub plus the thickness of the tape times the number of revolutions the hub has undergone. Substituting into the first equation we have

$$ds = \left(r_0 + \frac{\theta}{2\pi} T \right) d\theta$$

Integrating the left and right sides from 0 to *s* and 0 to θ respectively gives:

$$s = r_0 \theta + \frac{T}{2\pi} \frac{\theta^2}{2}$$

We now have to make some substitutions to get this into the final form. The substitutions used are

$$s = vt$$
$$\theta = 2\pi n$$
$$n = \alpha c$$

Substituting these in one at a time and simplifying we get:

$$vt = r_0 \theta + \frac{T}{2\pi} \frac{\theta^2}{2}$$
$$vt = r_0 2\pi n + \frac{T}{2\pi} \frac{(2\pi n)^2}{2}$$
$$vt = r_0 2\pi n + \frac{T}{4\pi} 4\pi^2 n^2$$
$$vt = r_0 2\pi n + T\pi n^2$$
$$vt = r_0 2\pi \alpha c + T\pi \alpha^2 c^2$$
$$t = \frac{2\pi r_0 \alpha}{v} c + \frac{T\pi \alpha^2}{v} c^2$$

Explanation of TAPINT

Given our equation $t = Ac + Bc^2$, we can solve for A and B if we know 2 time-counter reading pairs that satisfy the equation as follows:

We know that

$$t_1 = Ac_1 + Bc_1^2$$
$$t_2 = Ac_2 + Bc_2^2$$

If we let D= $\begin{vmatrix} c_1 & c_1^2 \\ c_2 & c_2^2 \end{vmatrix}$ then by Cramer's rule:

$$A = \frac{\begin{vmatrix} t_1 & c_1^2 \\ t_2 & c_2^2 \end{vmatrix}}{D}$$

and

$$B = \frac{\begin{vmatrix} c_1 & t_1 \\ c_2 & t_2 \end{vmatrix}}{D}$$

TAPINT uses these formulas.

AMERICAN JOURNAL OF EPIDEMIOLOGY
Vol. 123 No. 5

Printed in U.S.A.

Epidemiologic Programs for Computers and Calculators

SIMPLE ALGORITHMS FOR THE REPRESENTATION OF DETERMINISTIC AND STOCHASTIC VERSIONS OF THE REED-FROST EPIDEMIC MODEL USING A PROGRAMMABLE CALCULATOR

EDUARDO L. F. FRANCO[1] AND A. RAY SIMONS[2]

Franco, E. L. F. (Ludwig Institute for Cancer Research, São Paulo, SP 01509, Brazil) and A. R. Simons. Simple algorithms for the representation of deterministic and stochastic versions of the Reed-Frost epidemic model using a programmable calculator. *Am J Epidemiol* 1986;123:905–915.

Two programs are described for the emulation of the dynamics of Reed-Frost progressive epidemics in a handheld programmable calculator (HP-41C series). The programs provide a complete record of cases, susceptibles, and immunes at each epidemic period using either the deterministic formulation or the trough analogue of the mechanical model for the stochastic version. Both programs can compute epidemics that include a constant rate of influx or outflux of susceptibles and single or double infectivity time periods.

epidemiologic methods; disease outbreaks; models, theoretical; probability

One of the most powerful models used to illustrate the dynamics of infectious disease epidemics was originally developed by Lowell J. Reed and Wade Hampton Frost at The Johns Hopkins University School of Hygiene and Public Health (1). This model was derived from the one originally proposed by Soper (2) and has been used extensively as a teaching and research tool in epidemiology (3). Although stringent in algebraic formulation, the Reed-Frost model is versatile enough to represent theoretical progressive epidemics in deterministic or stochastic versions.

The deterministic formulation provides a method for stepwise calculation of the number of new cases of a disease at successive time intervals. Given initial conditions for the spread of the disease, e.g., number of susceptible individuals and index cases, and the probability of infective contact between any two individuals in the population, it is possible to calculate recursively the frequencies of new cases and susceptibles remaining during all subsequent time periods (1). The stochastic version illustrates the effects of the random variation that exists among epidemic trials performed sequentially and is intended to approach in fit the types of epidemic curves seen in natural settings (4).

Perhaps the most remarkable aspect of the Reed-Frost epidemic theory was the development of mechanical analogues for the stochastic formulation. Horiuchi and Sugiyama (5) proposed one such method which used a Monte Carlo approach based on shuffled "chips" to randomize their epidemic trials. However, the simplest and most didactic method was the one described

Received for publication June 24, 1985, and in final form September 20, 1985.

[1] Ludwig Institute for Cancer Research, São Paulo Branch, Rua Prof. Antonio Prudente, 109, São Paulo, SP 01509, Brazil. (Reprint requests to Dr. Eduardo L. F. Franco.)

[2] Division of Laboratory Training and Consultation, Centers for Disease Control, Atlanta, GA.

The authors are indebted to Bernard Greenberg, School of Public Health, University of North Carolina, for his guidance.

by Elveback and Varma (6). In this method (credited by the authors to Reed and Frost) the epidemic process can be visualized and measured through the random alignment of colored balls into a narrow trough. Each color corresponds to the individual's status during each epidemic cycle, i.e., whether he was a case, a susceptible, or an immune. In addition to these types, the method employed balls of a fourth color to represent boundaries (blocks) of effective contact among individuals. The probability of effective contact between any two individuals was a function of the number of these blocks which could be varied as desired. Each epidemic period is drawn by pouring the previously randomized population of balls into the trough. All "susceptible" balls falling in segments between blocks containing at least one "case" ball will be considered to have had effective "infectious" contact and, for the next period, they will be replaced by an equal number of "case" balls. Likewise, "case" balls during each period become "immune" balls for the next one, losing their "infective" nature. After all replacements have been made, the process is repeated, i.e., randomizing, pouring, and counting, for each subsequent period.

This trough method has been the mechanical analogue of choice for empirical samplings of the stochastic Reed-Frost epidemic process. Nevertheless, despite being didactic in principle, the trough method may be time-consuming and error-prone when used for multiple samplings. Repeated trials are best accomplished using a computer through programs that generate random numbers to simulate binomial trials (6). Use of microcomputers enhances tremendously the usefulness of the model, enabling classroom or field demonstration. Therefore, we have developed simple algorithms for computation of deterministic and stochastic Reed-Frost epidemics which can be used with a handheld programmable calculator. In its stochastic version the actual trough method is emulated pictorially, allowing a better appraisal of contacts made during each epidemic period.

MATERIALS AND METHODS

Equipment

Programs were written for an HP-41CV programmable calculator (Hewlett-Packard Co., Corvallis, OR). The minimal system configurations enabling execution of the two programs are 1) the basic HP-41C calculator equipped with two HP-82106A memory modules or one HP-82170A Quad memory module, or 2) an HP-41CV or an HP-41CX calculator. Use of the HP-82182A time module enables generation of automatic time-dependent random number seeds. Although this would have represented a desirable feature for the stochastic version, we resorted to a keyboard-entered, user-chosen seed to reset the generator. The programs contain several instructions to allow printing of the output. We used an HP-82143A printer (Hewlett-Packard Co.) to obtain a permanent record of the results, i.e., frequencies of cases, susceptibles, and immunes at each epidemic period and a pictorial representation of the aligned balls in the trough. However, use of a printer is not mandatory since all results, except the trough picture, will be displayed in the calculator also. Unattended program execution is possible only with the printer present in the system. Use of the HP-82104A card reader is optional.

Model formulations

The basic assumptions for the Reed-Frost model were adapted by Abbey (4) from Frost (7). (Frost's statements had been divulged in 1928 but were published only in 1976.) They were maintained in the present method with the minor nonrestrictive modifications of allowing two different infectivity times and influx or outflux of susceptibles into the population. The conventional notation for the deterministic and stochastic equations uses a common set of variables. S_t, C_t, and I_t are the numbers of susceptibles, cases, and immunes, respectively, in the population during epidemic time period t (the next period is represented as $t + 1$, and the preceding one as $t - 1$); p, or its complement, $1 - q$, represents the binomial probability that any two randomly selected individuals from the population will have effective contact during one time period.

Using such a notation, the deterministic formulation of the model can be summarized with three equations:

$$C_{t+1} = S_t(1 - q^{C_t}) \quad (1)$$

$$S_{t+1} = S_t - C_{t+1} \quad (2)$$

$$I_{t+1} = I_t + C_t \quad (3)$$

Enhancements were added to the above set of equations to enable simulation of situations where cases may be infective for an additional period before becoming immune and, as a consequence with the present algorithm, idle with respect to the dynamics of the epidemic. In addition, the present algorithms permit a constant influx or outflux of susceptibles into the population to simulate the effects of migration. With these enhancements the deterministic equations become:

$$C_{t+1} = S_t(1 - q^{C_t})$$

(1a) if infectivity time = 1

$$C_{t+1} = S_t(1 - q^{C_t + C_{t-1}})$$

(1b) if infectivity time = 2

$$S_{t+1} = S_t - C_{t+1} + S_f$$

(2a)

$$I_{t+1} = I_t + C_t$$

(3a) if infectivity time = 1

$$I_{t+1} = I_t + C_{t-1}$$

(3b) if infectivity time = 2

where S_f is the influx or outflux (when negative) of susceptibles in the population and newly computed C values are keyed into the other expressions only after being rounded to its nearest integer.

Using the same convention, the stochastic formulation (4, 8) is represented by the binomial probabilistic equation:

$$\text{Prob}(C_{t+1}) = [S_t!(1 - q^{C_t})^k (q^{C_t})^{S_t - k}]/[k!(S_t - k)!]$$

where $k = C_{t+1}$ and thus, $S_t - k = S_{t+1}$.

Iterative computation of S_{t+1} and I_{t+1} with respect to C_t, C_{t+1}, and S_f is done as above for the deterministic formulation.

For the trough analogue the number of blocks b can be calculated (8) from a given infectivity value p as follows: $p = (1 - q) = 2/(b + 2)$ and solving for b; $b = (2/p) - 2$.

Fine (8) pointed out that the basic deterministic formulation does not provide a suitable estimator for the expected number of cases to be obtained with the stochastic trough device. This is because the latter mechanism tends to underestimate the probability that a susceptible contacts at least one of many cases. Therefore, in order to control for this bias, a second formula for the deterministic version was utilized whenever average observed stochastic epidemics had to be compared with the expected results from the deterministic formulation. The new equation, as suggested by Fine (8), is obtained by replacing the term q^{C_t} in the above expressions 1 and 1a by:

$$b(b + 1)/[(b + C_t)(b + C_t + 1)] \quad (4)$$

Description of the algorithms

The two programs, REED and FROST, although totally independent with respect to register usage, flag control, and execution, share the same basic structure. Both programs are divided into two major routines: input and data processing. Numerical values for input in both programs are p, C_0, S_0, S_f, and infectivity time. Discrimination by the programs of nonzero S_f values and different infectivity times is achieved through flag control. In the stochastic version the program has an additional prompt for a seed to be keyed into the pseudorandom number generator. Computation of C, S, and I is straightforward in the deterministic version, REED. Computation of C values in each period through the stochastic version, FROST, is done as an emulation of the balls-and-trough method. Instead of color-coded balls the program collates the letters C, S, and I, and the symbol "*" (asterisk) to represent the trough alignment of cases, susceptibles, immunes, and

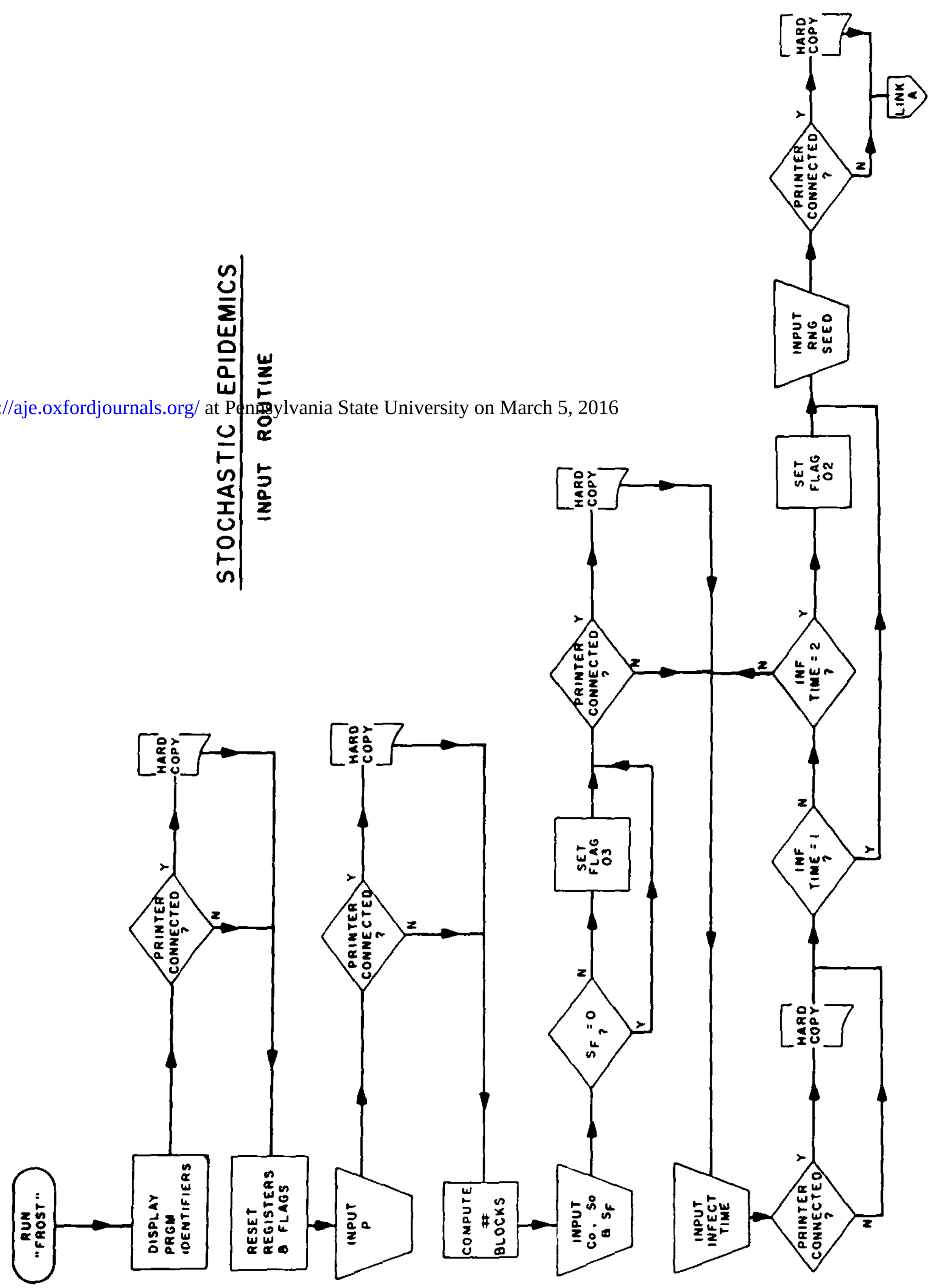


FIGURE 1. Flow diagram representing the input routine for program "FROST" which generates stochastic epidemics through the trough analogue of the mechanical model of the Reed-Frost epidemic formulation.

block balls, respectively. When the printer is connected to the system the trough sequence is represented in successive segments of 24 balls each. Each ball is drawn independently using a Monte Carlo approach through random sampling without replacement from four registers containing the frequencies b, $b + C$, $b + C + S$, and $b + C + S + I$. Generation of random numbers is done with a previously described device (9):

$$r_{i+1} = \text{Int}[s_{i+1}(b + C + S + I)] + 1$$

where

$$s_i = \text{Frac}(9{,}821 \times s_{i-1} + 0.211367).$$

The algorithms for the above routines in flow chart form are presented in figures 1 and 2.

Results

Program operation

Complete listings of the programs are provided in the appendix. Magnetic card copies of the two programs can be furnished by either author (requests should be accompanied by five blank cards). The operating instructions for each program are given in table 1. Program REED occupies 300 bytes of memory, requiring a size configuration of 007 to be executed. Program FROST is longer, taking 482 bytes and a size of 015. Computation of deterministic epidemics (program REED) is fast, taking less than 10 seconds per epidemic period. This contrasts with the several minutes to a few hours which it may take to complete certain stochastic epidemics presenting nonzero S_f

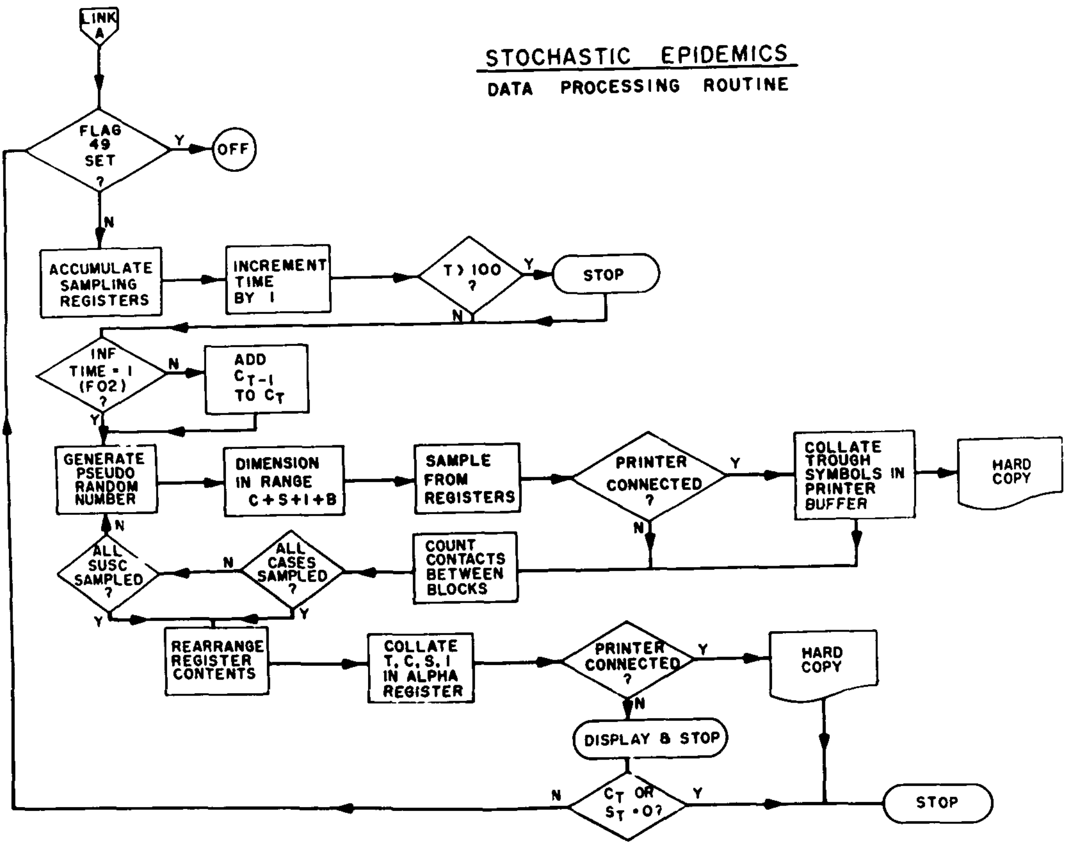


FIGURE 2. Flow diagram representing the data processing routine for program "FROST" which generates stochastic epidemics through the trough analogue of the mechanical model of the Reed-Frost epidemic formulation.

TABLE 1

Operating instructions for programs REED and FROST in user mode

Program step	Procedure	Input*	Output
	REED		
1	Load and run program (deterministic)		Identifiers
2	Entry of epidemic startup values as prompted in the display		
	Enter p	p	
	Enter C	C	
	Enter S	S	
	Enter S_f	S_f	
	Enter infectivity time	IT	Sets flag 00 if $IT = 1$
3	Computation of frequencies at each period. If printer is not attached, program interrupts at the end of each period (press R/S to resume)		T, C, S, and I; I is furnished only if $S_f = 0$
4	Execution terminates when $C = 0$. For a new set of data, key A and proceed with step 2 above. To reset press B		End of epidemic
	FROST		
1	Load and run program (stochastic)		Identifiers
2	Entry of epidemic startup values as prompted in the display		
	Enter p	p	
	Enter C	C	
	Enter S	S	
	Enter S_f	S_f	Sets flag 03 if $S_f \neq 0$
	Enter infectivity time	IT	Sets flag 02 if $IT = 2$
	Enter seed (≤9,999)	Seed	
3	Filling of the trough with balls according to random sampling. If printer is present, trough segments are formed.		⟨Trough⟩
	Computation of frequencies at each period. If printer is not attached, program interrupts at the end of each period (press R/S to resume)		T, C, S, and I; I is furnished only if $S_f = 0$
4	Execution terminates when $C = 0$ or $S = 0$. Proceed as with step 4 above.		End of epidemic

* R/S must be pressed after values are keyed in.

values (a low battery condition will interrupt execution because flag 49 is tested at the beginning of every period). If the thermal paper printer is connected to the calculator, it must be turned on to avoid interruption of program execution.

Epidemic simulations

Figure 3 shows three examples of printouts resulting from the execution of the deterministic epidemic program, REED, with startup conditions of $C = 1$, $S = 100$, $S_f = 0$, and infectivity time of 1. Printouts represent results with the above set of conditions for each of three p values, 0.02, 0.03, and 0.04. Figure 4 shows a complete epidemic using one of the above set of conditions ($p = 0.04$) but performed with the stochastic model program, FROST. As discussed, the letters C, S, and I, and the asterisk represent case, susceptible, immune, and block balls, respectively. The

```
REED-FROST EPIDEMICS
<DETERMINISTIC>

P? =0.020
C0? =1
S0? =100
S. INFLUX? =0
INF. TIME? =1

T=1  C=2  S=98  I=1
T=2  C=4  S=94  I=3
T=3  C=7  S=87  I=7
T=4  C=11  S=76  I=14
T=5  C=15  S=61  I=25
T=6  C=16  S=45  I=40
T=7  C=12  S=33  I=56
T=8  C=7  S=26  I=68
T=9  C=3  S=23  I=75
T=10  C=1  S=22  I=78
T=11  C=0  S=22  I=79
END OF EPIDEMIC
```

```
REED-FROST EPIDEMICS
<DETERMINISTIC>

P? =0.030
C0? =1
S0? =100
S. INFLUX? =0
INF. TIME? =1

T=1  C=3  S=97  I=1
T=2  C=8  S=89  I=4
T=3  C=19  S=70  I=12
T=4  C=31  S=39  I=31
T=5  C=24  S=15  I=62
T=6  C=8  S=7  I=86
T=7  C=2  S=5  I=94
T=8  C=0  S=5  I=96
END OF EPIDEMIC
```

```
REED-FROST EPIDEMICS
<DETERMINISTIC>

P? =0.040
C0? =1
S0? =100
S. INFLUX? =0
INF. TIME? =1

T=1  C=4  S=96  I=1
T=2  C=14  S=82  I=5
T=3  C=36  S=46  I=19
T=4  C=35  S=11  I=55
T=5  C=8  S=3  I=90
T=6  C=1  S=2  I=98
T=7  C=0  S=2  I=99
END OF EPIDEMIC
```

FIGURE 3. Three examples of printed output from program "REED" which is based on the deterministic formulation. The three epidemics displayed differed in the magnitude of the p values: 0.02, 0.03, and 0.04, respectively.

"<" and ">" symbols represent the left and right extremities of the trough, respectively. It is noteworthy that the printing of the trough is not necessarily completed at every period, e.g., $T = 1$ and $T = 6$ in figure 4. This is because the forwarding balance of each period is immediately calculated as soon as a block is aligned to the right of a segment containing the last drawn C or S ball.

Table 2 presents the numbers of cases produced at each epidemic period for 25 calculator-generated trough stochastic trials (listed in ascending order of duration). The combination of startup conditions chosen, $p = 0.04$, $C = 1$, $S = 100$, $S_f = 0$, and infectivity time of 1, was also used to generate two deterministic sequences, one by the general term formula (equation 1a), and the other with the alternative equation based on the number of blocks b (using expression 4 as replacement term). The observed mean stochastic epidemics did not differ significantly ($\chi^2 = 2.47$, $p = 0.4807$) from the expected deterministic epidemics generated by the latter process. However, the observed mean frequencies of cases at all periods were significantly different from the expected set of values computed by the original general term (equation 1a) deterministic formula ($\chi^2 = 8.82$, $p = 0.0318$). This is in agreement with the study on the mechanical analogue done by Fine (8) who suggested the modified formula as a better estimator of the expected number of cases obtained with the trough model.

DISCUSSION

The low speed of execution of program FROST is certainly a drawback if the user intends to run repeated trials of stochastic epidemics. This problem is even more pronounced with very low "infectivity" p values, which implies that a very high number of blocks (b) must be computed for sampling. Likewise, larger population sizes at the onset of the epidemics contribute to increase execution time for the same reason, i.e., by increasing the number of passes

```
REED-FROST EPIDEMICS
<STOCHASTIC><TROUGH>

P? =0.040
C0? =1
S0? =100
S. INFLUX? =0
INF. TIME? =1
SEED<=9999?=7460

<S**SS*S*SSS**SSSSSSSSSC
SSSSSS*

T=1  C=15  S=85  I=1

<*S**SS*S**CSS*SSSI*S*C*
SCS*SCSCSSS*S*SSSSSS*S*C
SSSS**S*SSSSSSS**SSCSS*S
S*SC*SS**CSSS*SS*SS*S*S*
S*S*SSS**SCSS*CSSSS*S*SS
SCCSSSSSC*SSSS**S*S**S*S
****SC>

T=2  C=37  S=48  I=16

<S*I*I*IIS***SSCI**C****
*SCCI*SS*CS*SSI*IISISSS*
**CS*CS*CC*C*S*S**SICC**
SSC*CSCCC**SICIS*CS*S*SS
C*SSS*S***CCICICSSCSCCCS
*C*C*SCSSSIS***SSCS*C*SS
CCCSSC>

T=3  C=31  S=17  I=53

<C*CI***IIII**S*I****C*I
CCSIII*IIC*IIC***I*I*CC*
*CSIICCICCSS*IIII*II*CI*
IICII**IIISI**III**CCCII
*SSSC*SCISIII*CC**IICIII
SS*CCC**S*CCIII*C*******
SSICSI>

T=4  C=14  S=3  I=84

<*II*I***II*ICIIC*I*III*
ICIII**I*II*I*IIICICIS*I
*II***C*I**I*IIII*I****I
II**IIIII*CIIICI*IIIII*S
IIIIC**IC*IISI***IICCII*
III*III**I*III*III*I*III
*CICII>

T=5  C=2  S=1  I=98

<I****II*IIII**IIIIIIII*
*IIII*****II*I**IIII*III
I*I**II***IIIIIIIII*II**
I****IIIC**IIIII*III*III
III*IIIIIIIIIS*CII*

T=6  C=0  S=1  I=100

END OF EPIDEMIC
```

FIGURE 4. Example of printed output from program "FROST" which is based on the stochastic formulation. The output emulates the trough model as a mechanical analogue where each successive period is composed of one or more "window" segments on the ball-filled trough and the resulting statistics for the next period.

through the Monte Carlo sampling device at each epidemic period. Reduction in execution time is possible by direct implementation of the binomial probabilistic equation since this would provide direct calculation of frequencies of cases, susceptibles, and immune at a single randomization pass. However, such a shortcut method would not be nearly illustrative of the dynamics of the epidemics as the balls-and-trough method which has a dramatic didactic appeal. Adaptation of the same algorithm for execution as a microcomputer BASIC program is another solution for the slow execution time caused by the multiple iterations for the computation of the trough sequence.

Although these algorithms are presented here as independent programs, they can be easily adapted to be used as subroutines of larger main programs. Such larger programs could be designed to include additional features such as computation of descriptive statistics and other values. With the increasing availability of lower cost and powerful microcomputers and programmable, interface-ready calculators, algorithms containing iterative Monte Carlo devices

TABLE 2

Numbers of cases generated in 25 trials of the calculator-emulated trough model mechanical analogue of the Reed-Frost stochastic epidemic formulation. Comparison with two deterministic formulations*

Seed† no.	Epidemic period									Size	Peak	Length
	1	2	3	4	5	6	7	8	9			
1,416	0									0	0	0
8,963	0									0	0	0
1,489	0									0	0	0
9,873	0									0	0	0
951	0									0	0	0
2,568	1	0								1	1	1
6,537	9	31	45	10	0					95	45	4
4,168	12	33	37	16	0					98	37	4
236	4	10	31	36	13	0				94	36	5
1,435	5	9	28	37	16	0				95	37	5
1,587	10	31	37	15	3	0				96	37	5
1,569	7	23	34	28	4	0				96	34	5
125	2	18	50	23	4	0				97	50	5
7,460	15	37	31	14	2	0				99	37	5
4,128	6	33	39	21	1	0				100	39	5
5,681	2	8	22	36	22	5	0			95	36	6
2,581	3	11	30	41	11	1	0			97	41	6
215	3	3	16	47	24	5	0			98	47	6
5,629	13	23	34	15	11	2	0			98	34	6
1,254	3	12	32	33	15	4	0			99	33	6
1,235	7	20	31	23	10	3	1	0		95	31	7
547	2	3	12	34	29	11	4	0		95	34	7
5	4	26	41	22	3	1	1	1	0	98	41	7
8,573	3	9	22	24	23	8	1	1	0	91	24	8
256	2	6	9	34	24	14	3	1	0	93	34	8
Mean	4.5	17.3	30.6	26.8	11.3	3.2	1.0	0.4	0.0	96.3	37.2	5.8
Original‡	4	14	36	35	8	1	0			98	36	6
Modified‡	4	14	33	32	11	2	0			96	33	6

* Based on the following conditions at period zero: $C = 1$, $S = 100$, $p = 0.04$, $S_f = 0$, and infectivity time of 1.

† Used to recycle pseudo random number generator before each trial.

‡ Results with two deterministic formulations: original (as in program REED) and modified (using expression 4) equations.

will become more useful in epidemiology. The one presented here was implemented in a fully portable system and has the desirable trough emulation feature which characterizes the most didactic mechanical analogue of the stochastic Reed-Frost epidemic model.

REFERENCES

1. Maia JOC. Some mathematical developments on the epidemic theory formulated by Reed and Frost. Hum Biol 1952;24:167-200.
2. Soper HE. Interpretation of periodicity in disease prevalence. J Roy Stat Soc 1927;92:34-73.
3. Sartwell PE. Memoir on the Reed-Frost epidemic theory. Am J Epidemiol 1976;103:138-40.
4. Abbey H. An examination of the Reed-Frost theory of epidemics. Hum Biol 1952;24:201-33.
5. Horiuchi K, Sugiyama H. On the importance of Monte Carlo approach in the research of epidemiology. Osaka City Med J 1957;4:59-62.
6. Elveback L, Varma A. Simulation of mathematical models for public health problems. Public Health Rep 1965;80:1067-76.
7. Frost WH. Some conceptions of epidemics in general. Am J Epidemiol 1976;103:141-51.
8. Fine PEM. A commentary on the mechanical analogue to the Reed-Frost epidemic model. Am J Epidemiol 1977;106:87-100.
9. Heyman VK. Random number generators. 65 Notes 1977;8:1-6.

APPENDIX: PROGRAM LISTINGS

```
 01♦LBL "REE
D"
 02 "REED-FR
OST"
 03 "⊢ EPIDE
MICS"
 04 AVIEW
 05 "<DETERM
INISTIC>"
 06 AVIEW
 07 ADV
 08♦LBL A
 09 1 E-1
 10 STO 04
 11 CLX
 12 STO 05
 13 STO 06
 14 CF 00
 15 FIX 3
 16 1
 17 TONE 9
 18 "P? ="
 19 PROMPT
 20 -
 21 STO 00
 22 ARCL L
 23 FS? 55
 24 PRA
 25 CF 29
 26 FIX 0
 27 TONE 9
 28 "C0? ="
 29 PROMPT
 30 STO 02
 31 ARCL X
 32 FS? 55
 33 PRA
 34 TONE 9
 35 "S0? ="
 36 PROMPT
 37 STO 03
 38 ARCL X
 39 FS? 55
 40 PRA
 41 TONE 9
 42 "S. INFL
UX? ="
 43 PROMPT
 44 STO 01
 45 ARCL X
 46 FS? 55
 47 PRA
 48♦LBL 00
 49 TONE 9
 50 "INF. TI
ME? ="
 51 PROMPT
 52 ARCL X
 53 FS? 55
 54 PRA
 55 ADV
 56 2
 57 X=Y?
 58 GTO 01
 59 CLX
 60 1
 61 X≠Y?
 62 GTO 00
 63 SF 00
 64♦LBL 01
 65 ISG 04
 66 GTO 02
 67 TONE 9
 68 STOP
 69♦LBL 02
 70 RCL 00
 71 RCL 02
 72 FC? 00
 73 XEQ 03
 74 Y↑X
 75 1
 76 -
 77 CHS
 78 RCL 03
 79 *
 80 RND
 81 "T="
 82 ARCL 04
 83 "⊢  C="
 84 ARCL X
 85 RCL 06
 86 RCL 02
 87 FS? 00
 88 ST+ 05
 89 STO 06
 90 RDN
 91 FC? 00
 92 ST+ 05
 93 RDN
 94 ST- 03
 95 STO 02
 96 RCL 01
 97 ST+ 03
 98 "⊢  S="
 99 ARCL 03
100 X≠0?
101 GTO 04
102 "⊢  I="
103 ARCL 05
104♦LBL 04
105 FC? 55
106 PROMPT
107 FS? 55
108 PRA
109 RCL 02
110 X=0?
111 GTO 05
112 GTO 01
113♦LBL 03
114 RCL 06
115 +
116 RTN
117♦LBL 05
118 TONE 9
119 "END OF
EPIDEMIC"
120 AVIEW
121 ADV
122 RTN
123♦LBL B
124 CF 00
125 SF 29
126 FIX 2
127 END

LBL'REED
END
300 BYTES
```

REED

```
01◆LBL "FRO
ST"
 02 "REED-FR
OST"
 03 "⊢ EPIDE
MICS"
 04 AVIEW
 05 "<STOCHA
STIC>"
 06 "⊢<TROUG
H>"
 07 AVIEW
 08 ADV
09◆LBL A
 10 1 E-1
 11 STO 13
 12 CLX
 13 STO 07
 14 STO 09
 15 STO 10
 16 STO 14
 17 CF 00
 18 CF 01
 19 CF 02
 20 CF 03
 21 FIX 3
 22 2
 23 TONE 9
 24 "P? ="
 25 PROMPT
 26 ARCL X
 27 FS? 55
 28 PRA
 29 CF 29
 30 FIX 0
 31 /
 32 2
 33 -
 34 RND
 35 STO 04
 36 TONE 9
 37 "C0? ="
 38 PROMPT
 39 STO 06
 40 ARCL X
 41 FS? 55
 42 PRA
 43 TONE 9
 44 "S0? ="
 45 PROMPT
 46 STO 05
 47 ARCL X
 48 FS? 55
 49 PRA
 50 TONE 9
 51 "S. INFL
UX? ="
 52 PROMPT
 53 STO 12
 54 ARCL X
 55 X≠0?
 56 SF 03
 57 FS? 55
 58 PRA
 59 TONE 9
 60◆LBL 04
 61 "INF. TI
ME? ="
 62 PROMPT
 63 ARCL X
 64 FS? 55
 65 PRA
 66 1
 67 X=Y?
 68 GTO 05
 69 CLX
 70 2
 71 X≠Y?
 72 GTO 04
 73 SF 02
 74◆LBL 05
 75 TONE 9
 76 "SEED<=9
999?="
 77 PROMPT
 78 ARCL X
 79 FS? 55
 80 PRA
 81 ADV
 82 1 E4
 83 /
 84 STO 08
 85◆LBL 06
 86 FS? 49
 87 OFF
 88 "<"
 89 FS? 55
 90 ACA
 91 RCL 07
 92 RCL 06
 93 RCL 05
 94 RCL 04
 95 STO 00
 96 +
 97 STO 01
 98 +
 99 STO 02
100 FC? 03
101 +
102 STO 03
103 ISG 13
104 GTO 08
105 BEEP
106 STOP
107◆LBL 08
108 FC? 02
109 GTO 07
110 RCL 14
111 ST+ 02
112 ST+ 03
113◆LBL 07
114 .003
115 STO 11
116 RCL 08
117 9821
118 *
119 .211327
120 +
121 FRC
122 STO 08
123 RCL 03
124 *
125 INT
126 1
127 +
128◆LBL 09
129 RCL IND
11
130 X<>Y
131 X<=Y?
132 GTO IND
11
133 ISG 11
134 GTO 09
135 ">"
136 FS? 55
137 ACA
138 GTO 13
139◆LBL 00
140 "*"
141 SF 01
142 1
143 GTO 10
144◆LBL 01
145 "S"
146 1
147 ST+ 10
148 GTO 10
149◆LBL 02
150 "C"
151 SF 00
152 1
153 GTO 10
154◆LBL 03
155 "I"
156 1
157◆LBL 10
158 ST- IND
11
159 ISG 11
160 GTO 10
161 FS? 55
162 ACA
163◆LBL 11
164 FC?C 01
165 GTO 07
166◆LBL 13
167 FC?C 00
168 GTO 12
169 RCL 10
170 ST+ 09
171◆LBL 12
172 0
173 STO 10
174 RCL 01
175 RCL 02
176 X>Y?
177 GTO 07
178 RCL 09
179 RCL 14
180 PCL 06
181 FC? 02
182 ST+ 07
183 STO 14
184 RDN
185 FS? 02
186 ST+ 07
187 RDN
188 ST- 05
189 STO 06
190 RCL 12
191 ST+ 05
192 0
193 STO 09
194 BEEP
195 FS? 55
196 PRBUF
197 ADV
198 "T="
199 ARCL 13
200 "⊢ C="
201 ARCL 06
202 "⊢ S="
203 ARCL 05
204 FS? 03
205 GTO 14
206 "⊢ I="
207 ARCL 07
208◆LBL 14
209 AVIEW
210 FC? 55
211 STOP
212 CLD
213 ADV
214 RCL 06
215 X>0?
216 RCL 05
217 X>0?
218 GTO 06
219 TONE 9
220 "END OF
EPIDEMIC"
221 AVIEW
222 ADV
223 RTN
224◆LBL B
225 CF 00
226 CF 01
227 CF 02
228 CF 03
229 SF 29
230 FIX 3
231 END

LBL'FROST
END
482 BYTES

SIZE = 015
```

FROST