

TK Solver – A Tool Kit of Many Solvers!

TK Solver is a “Tool Kit” full of many solvers.

Direct Solver
Iterative Solver
Procedure Processor
Differential Equations Integrator
List Solver
Optimizer
TK Library of Root-Finders
Excel Tool Kit

Here is a summary with examples.

TK's *Direct Solver* performs inverse mathematical operations and inverts functions in an equation to solve for a single unknown. Then it applies what it learns to subsequent equations, making additional passes through the rules to solve previously bypassed equations and checking for inconsistencies.

For example, given inputs for any two of the three variables in the following equation, TK's *Direct Solver* will solve for the remaining variable. $\text{Tand}(\text{theta}/2) = \text{radius}/\text{height}$, or in MathLook form,

$$\text{tand}\left[\frac{\text{theta}}{2}\right] = \frac{\text{radius}}{\text{height}}$$

The *Direct Solver* makes your life easier by allowing you to enter your equations in any sequence and without having to isolate the unknowns to left of the equals sign.

TK's *Iterative Solver* works with the *Direct Solver* to solve equations where the unknown cannot be isolated or to solve systems of simultaneous equations. The *Iterative Solver* sends a guess value to the *Direct Solver*, which works its way through the equations until it detects an inconsistency in one of the equations. The *Direct Solver* then tells the *Iterative Solver* how much of an inconsistency it found. The *Iterative Solver* keeps sending the *Direct Solver* new guesses until the inconsistency is very close to zero. The *Iterative Solver* can work with multiple guess variables if necessary.

For example, the following set of equations can be solved using TK's *Iterative Solver*, if inputs are given for theta and volume.

$$\text{tand}\left[\frac{\text{theta}}{2}\right] = \frac{\text{radius}}{\text{height}}$$
$$\text{volume} = \frac{1}{3} \cdot \pi \cdot \text{radius}^2 \cdot \text{height}$$

The *Iterative Solver* can also be used to solve problems involving more complex functions that are otherwise non-invertible. For example, the *Iterative Solver* can be used to determine the upper limit (x2) of the following integral, given the lower limit (x1), such that a desired area is achieved.

$$\text{Area} = \text{Integral}("1-\sin(x)*\cos(x),x",x1,x2)$$

Similarly, the *Iterative Solver* can backsolve the following rule to determine the pressure at which water vapor, at a given temperature, will have a given density.

$$\text{Rho} = \$\text{density}(1, \text{'water'}, \text{'T,T'}, \text{'P,P'})$$

TK Solver also includes a *Procedure Language* that allows for solving problems in a sequence of steps. The procedure language is used to create user-defined subroutines called Procedure Functions. Procedure Functions can be called as a subroutine by the Direct Solver to do specific tasks such as stepping through a range of inputs, filling lists, performing summations or products, etc. Procedure functions allow you to add any algorithms to a TK model. Given an input value for C, the following procedure loops through a series of calculations, returning the final value of S and generating a list called XYZ with the cosines of each of the 100 intermediate values of s.

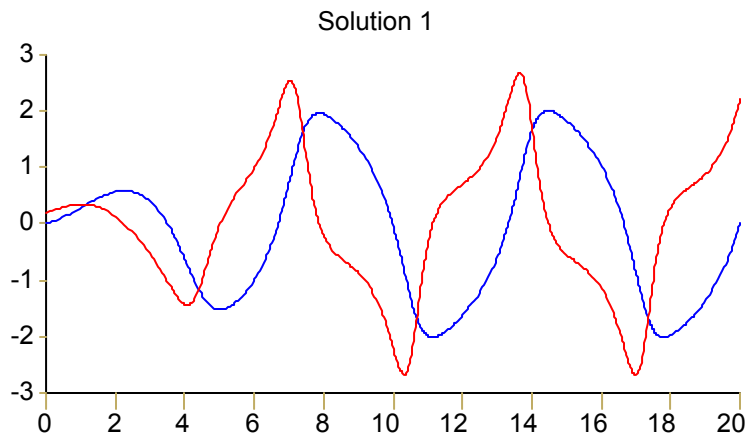
```
S = 0
For I = 1 to 100
  S = S + (C-I/1000)^-2
  'XYZ[I] = cos(S)
Next I
```

Suppose we need the output of this procedure to be 2.75 and we need to know the value C required to make that happen. The Iterative Solver handles it easily.

Some TK built-in functions automatically generate lists of solutions. For example, TK includes a collection of integrators for *solving differential equations*. There is also a Wizard to help users set up the differential equations, initial conditions, solution tables, and plots. As with procedure functions, the Iterative Solver can be used to backsolve these integrators to achieve desired results.

Here is TK's solution to the following differential equations, where $y_1(0) = 0$ and $y_2(0) = 0.2$

$$\begin{aligned} y_1' &= y_2 \\ y_2' &= \begin{bmatrix} 1 - y_1 & 2 \\ 2 & -y_1 \end{bmatrix} \cdot y_2 - y_1 \end{aligned}$$



Element	ODE_x1	ODE_y11	ODE_y12
1	0	0	.2
2	.092603848	.019377624	.21849121
3	.187649488	.041039322	.237275272
4	.282441726	.064403466	.255589567
5	.377411345	.089522341	.273262503
6	.472887231	.116423086	.290057758
7	.569168717	.145114274	.305677834

.....

Element	ODE_x1	ODE_y11	ODE_y12
541	19.8392069	-29388033	1.86533685
542	19.8694777	-23649045	1.92670722
543	19.8986642	-17937409	1.98741169
544	19.9267374	-12274748	2.04697566
545	19.953833	-.066495	2.10527704
546	19.9800655	-.01052269	2.16216518
547	20	.0330106	2.20546338

TK also includes a *List Solver* that will repeatedly run the entire math model, using lists of values for any of the input variables. This is one way to generate tables and plots of results. The List Solver can be set up manually or through the use of a Wizard. The List Solver will call on the Direct and Iterative Solvers at each step of the process. Here is set of TK rules used to solve fluid flow problems.

density = \$DENSITY(3, fluid, 'T', 'P', P)

viscosity = \$VISCOSITY(3, fluid, 'T', 'P', P)

$$q = \frac{w}{\text{density}}$$

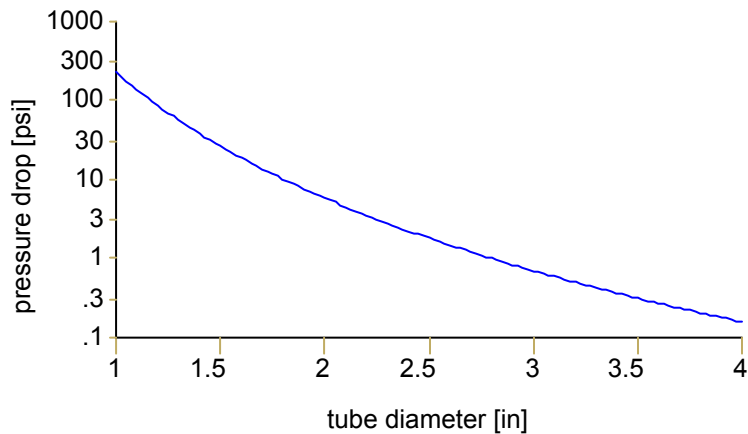
$$V = \frac{4 \cdot q}{\pi \cdot D^2}$$

$$\text{NRe} = \frac{V \cdot D \cdot \text{density}}{\text{viscosity}}$$

$$\frac{1}{\sqrt{f}} = -4 \cdot \log \left[\frac{\text{eps}}{44.4 \cdot D} + \left[\frac{1.255}{\text{NRe} \cdot \sqrt{f}} \right] \right]$$

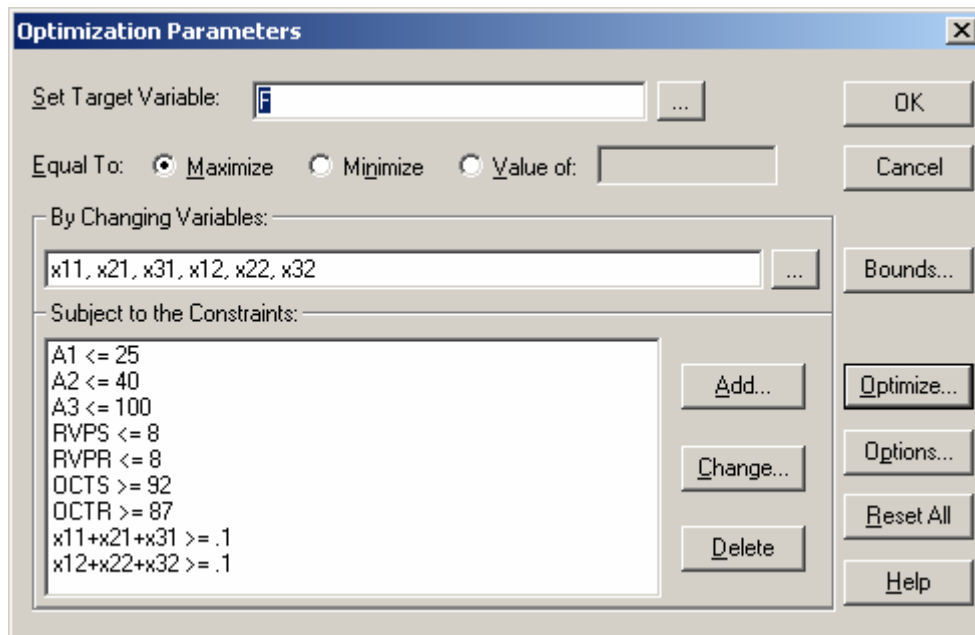
$$\frac{\text{deltap}}{L} = \frac{f \cdot \text{density} \cdot V^2}{D \cdot 16.085}$$

Here is a plot of the pressure drop vs. the tube diameter.



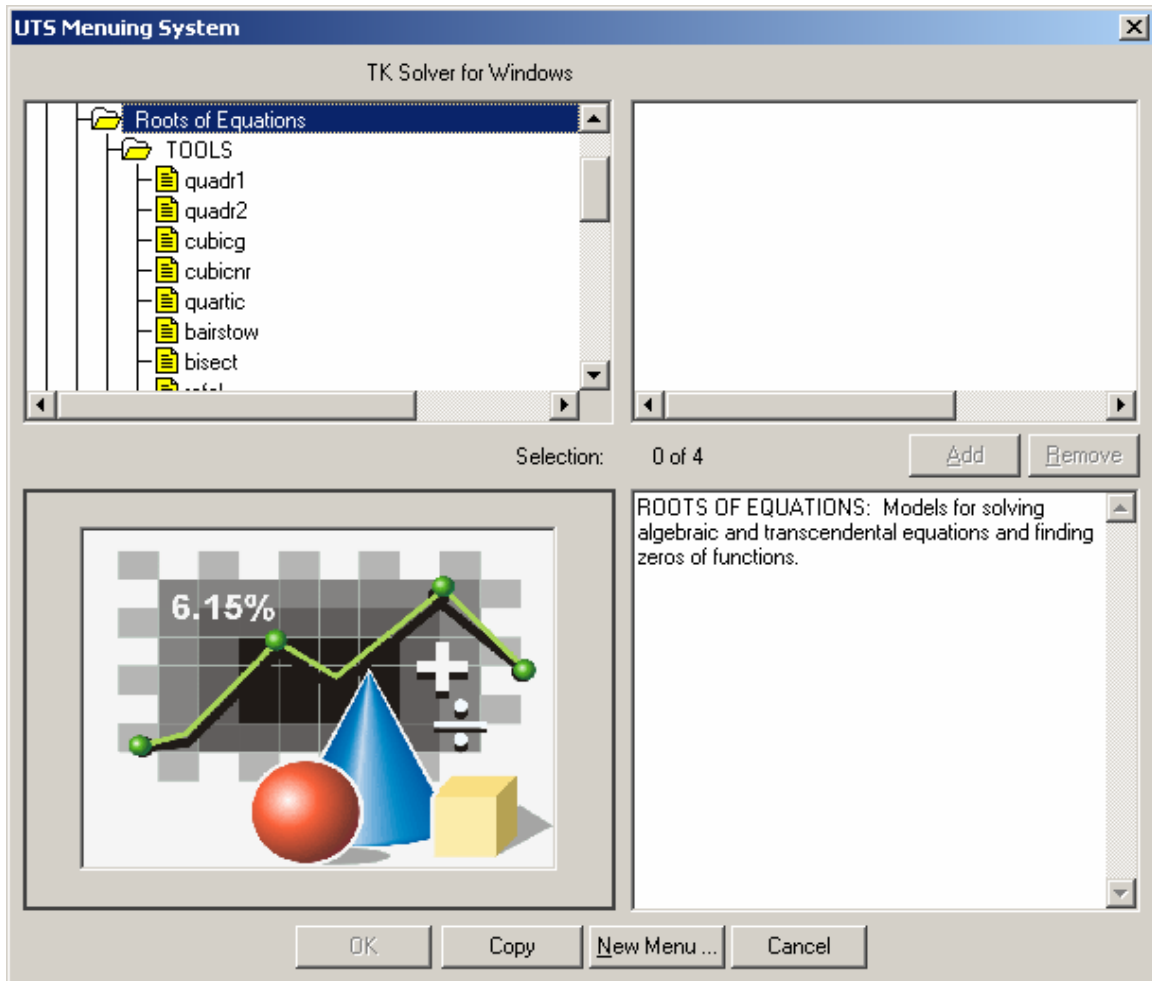
List Solving can also be used to run through a series of random numbers from various probability distributions. This is called Monte Carlo Simulation. TK's Simulation Wizard can be used to set up all the required parameters and to evaluate the results.

The TK *Optimizer* repeatedly solves a TK model until a set of objectives are met. This can involve finding a minimum or maximum value subject to sets of bounds and constraints on any of the variables in the model. The Optimizer can call on the Direct and Iterative Solvers at each step of the process.



The Optimizer can also be used as an alternative to the Iterative Solver for backsolving a model to determine how an input must be changed so that one of the outputs results in a desired value.

TK Solver includes a Library of pre-programmed procedures you can merge into your own models. The Library includes hundreds of tools divided into various categories. These tools are accessed via a graphical menu.



The TK Library tools can be used to solve specific problems within a math model, freeing the other solvers to tackle other tasks. The Library tools also add to the available root-finders in TK. Here is a quick summary of the root-finders in the TK Library.

Quadratic Equations

Cubic Equations

Quartic Equations

Nth Order Polynomial Equations by Bairstow's Method

Bisection Method

False Position Method

Newton's Method

You can see these tools in action by loading the examples from the TK Library before merging in one of the tools.

Finally, the *TK-Excel Toolkit* provides another level of solving power. The TK-Excel Toolkit allows you to run any number of TK models as subroutines within an Excel spreadsheet. The TK cells can also be referenced from other cells in Excel. The outputs from one model can be linked to the inputs in other models. The linked TK models can be called to List Solve to generate tables of solutions that can also be linked with Excel. The Excel Solver add-in can be used to optimize the results from this entire combination. Other popular Excel Add-ins such as Crystal Ball can also interact with the TK variables and lists in this way.

