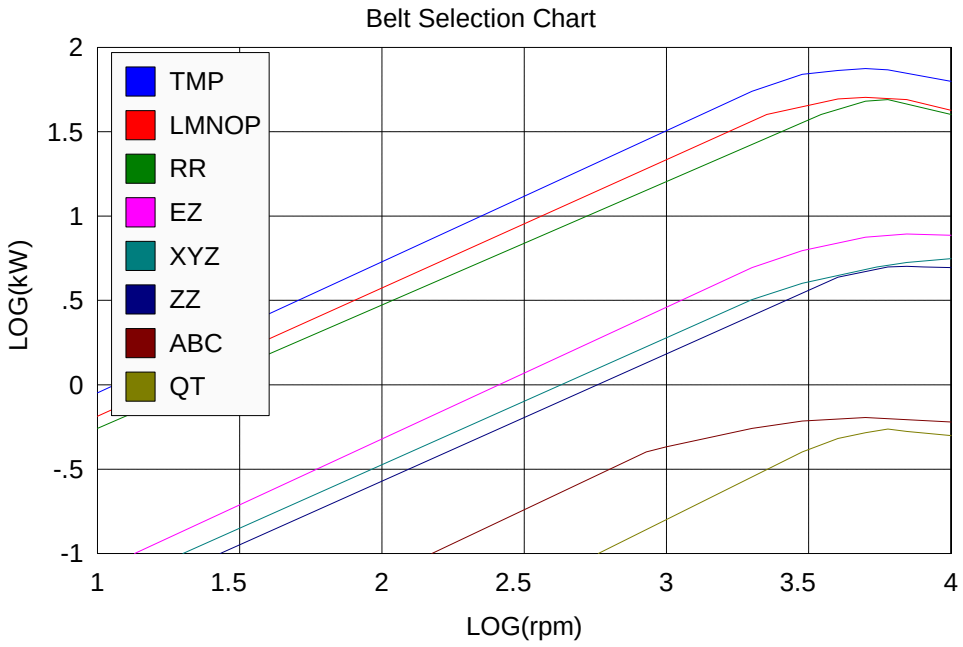
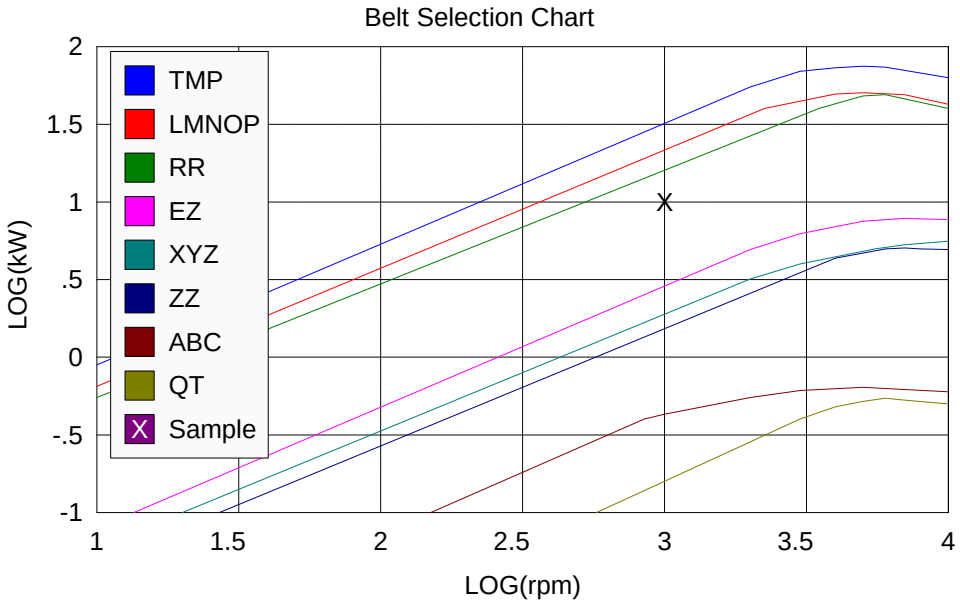
TK Solver Case Study: Applying Families of Functions in Product Selection Applications

Math models created for product selection or sizing frequently require the creation of families of performance functions. Given inputs defining the required performance, the model then repeatedly loops through the functions until the best product is found. There are several TK Solver features which make it ideal for such tasks – its abilities to work with text strings and to process functions with variable names.

Suppose a company manufactures eight different types of timing belts and they would like to build a math model to help choose the proper belt for any given application. Each belt might have a performance curve relating maximum power to pulley rpm. One of the selection criteria might then be to choose the belt whose power curve exceeds the required performance by the least amount.



This collection of curves above is plotted on log scales to make interpolation easier. The TMP belt has the maximum capacity and the QT belt the smallest. Suppose a belt is required for an application with a 10 kW motor driving a pulley at 1000 rpm. Which belt most closely exceeds the requirement? Using log(1000)=3 on the x-axis and log(10)=1 on the y-axis, we see the point marked by X just below the RR curve.



We can visually inspect the chart and see that the RR belt would work best based on this criterion. In practice, this may be just one of several criteria we need to check and confirm simultaneously. In such cases, visual interpretation is tedious, error prone, and inefficient. A math model can automate the task, eliminating these problems. A TK procedure function can do the job. The procedure will loop through the belt curves until one is found that exceeds the requirement.

First, a master list is created containing the names of each of the belts. The names are listed in reverse order of power capacity. That is, the belt with the lowest capacity is listed first and the one with the highest capacity is listed last. Here is the list named bd.

Element	Value
1	QT
2	ABC
3	ZZ
4	XYZ
5	EZ
6	RR
7	LMNOP
8	TMP

List functions are created for each of the belt performance curves. These selection functions are given names sABC, sXYZ, and so on with the character s preceding each of the belt names. The domain lists for these functions are named by adding a d to the end of the function name. The range lists have an r added to the end of the function name. As you will see, such consistency in naming the functions will pay off in programming efficiency.

Comment:	TMP power/rpm function
Domain List:	sTMPd
Mapping:	Linear
Range List:	sTMPr

Element	Domain	Range
1	1	-0.048
2	3.301	1.74
3	3.477	1.84
4	3.602	1.863
5	3.699	1.875
6	3.778	1.868
7	4	1.799

The looping procedure function can now be created to select the best belt. The procedure below assumes that the variables rpm and P are inputs and that the list functions have been set up as described above.

```
lP = log(P)       ; locate the y-axis coordinate
ln = log(rpm)     ; locate the x-axis coordinate
for i = 1 to 8    ; possibly check each of the 8 belt types
  sf = join('s,'bd[i])   ; build the function name for the belt
  sfd = join(sf,'d)      ; build the domain list name for the belt function
  if ln < min(sfd) then continue     ; if the speed is below the domain minimum for the belt, try the next belt
  Ps = apply(sf,ln)    ; compare value against the list function for the belt
  if lP > Ps then continue      ; keep looping until the given value falls below a curve, then choose that belt
  bdes = 'bd[i]   ; selection based on power requirement
  exit     ; once the selection is made, leave the loop and move on to other calculations
next i
```

The use of TK's APPLY function allows the name of the evaluated function to change as the procedure loops through the list. For example, the first time through the loop (i=1), the local variable sf is assigned the value 'sQT by the join function, APPLY then uses the sQT function and the value of ln to compute the

value Ps. The next time though the loop (i=2), sf = ‘sABC and APPLY processes the variable ln with the function sABC.

The rule and variable sheets are very simple.

Rule
belt = select(rpm,power)

Status	Input	Name	Output	Unit	Comment
					Belt Selection
	1000	rpm		rpm	RPM of smaller pulley
	10	power		kW	Power to be transmitted
		belt	RR		Belt type

UTS specializes in creation of web-based product selectors using math models like this one run by RuleMaster. We encourage other application developers to take advantage of these tools as well.