This Maple worksheet documents the calculations described in Section 4.2 for System 28 of the following paper.

Matthew England, Hassan Errami, Dima Grigoriev, Ovidiu Radulescu, Thomas Sturm, and Andreas Weber. Symbolic Versus Numerical Computation and Visualization of Parameter Regions for Multistationarity of Biological Networks. In Proceedings of CASC '17, Beijing, China, September 18 -22 2017, 15 pages. Springer, 2017.

Note - we are running in Maple 2016 with the latest version of the RegularChains linrary downloaded from www.regularchains.org

Date Written: 13th June 2017
Author: Matthew England (Coventry University)
Matthew.England@coventry.ac.uk

```
> restart:
> libname := "C:/Users/ab9797/Dropbox/Maple_CAD/FromRCorg/Latest",
  libname;
```

libname := "C:/Users/ab9797/Dropbox/Maple_CAD/FromRCorg/Latest",
"C:\Program Files\Maple 2016\lib"

```
> with(RegularChains):
> with(SemiAlgebraicSetTools);
```

[*BoxValues, Complement, CylindricalAlgebraicDecompose, Difference, DisplayParametricBox, DisplayQuantifierFreeFormula, EmptySemiAlgebraicSet, Intersection, IsContained, IsEmpty, IsParametricBox, LinearSolve, PartialCylindricalAlgebraicDecomposition, PositiveInequalities, Projection, QuantifierElimination, RealRootCounting, RealRootIsolate, RefineBox, RefineListBox, RemoveRedundantComponents, RepresentingBox, RepresentingChain, RepresentingQuantifierFreeFormula, RepresentingRootIndex, SignAtBox, VariableOrdering*]

Funnctions to iterate the grid sampling

```
> generate := proc(KaR, KbR, KcR, ansTab, TS, vars, update:=
  false, cadOnly:=false)
  local check, r, Kavals, Kbvals, Kcvals, ans, i:

  for r in [KaR, KbR, KcR] do
    check := (type(r, list) and nops(r)=3 and r[2]>r[1]) or type
  (r, integer):
    if check<>true then
      error("First 3 arguments must each be an integer or a list
  of 3 integers (start, stop, jump)"):
    fi:
  od:

  if type(ansTab, table)<>true then
    error("4th argument must be a table"):
  fi:

  if type(TS, list)<>true then
    error("5th argument must be a list of polynomial
  constraints"):
  fi:

  if type(vars, list)<>true then
    error("6th argument must be a list of variables"):
  fi:

  if not( type(update,integer) or is(update=false) ) then
    error("Optional 7th argument must be an integer")
  fi:

  if type(KaR, integer) then
      Kavals:= [KaR]:
  else
      Kavals := []:
      for i from KaR[1] by KaR[3] to KaR[2] do
        Kavals := [op(Kavals), i]:
      od:
  fi:

  if type(KbR, integer) then
      Kbvals:= [KbR]:
  else
      Kbvals := []:
      for i from KbR[1] by KbR[3] to KbR[2] do
        Kbvals := [op(Kbvals), i]:
      od:
  fi:

  if type(KcR, integer) then
      Kcvals:= [KcR]:
  else
      Kcvals := []:
```

```
    for i from KcR[1] by KcR[3] to KcR[2] do
      Kcvals := [op(Kcvals), i]:
    od:
fi:

if cadOnly=false then:
  generate_inner(Kavals, Kbvals, Kcvals, ansTab, TS, vars,
update):
else:
  generate_innerCO(Kavals, Kbvals, Kcvals, ansTab, TS, vars,
update):
fi:

return(0):

end proc:


generate_inner := proc(Kavals, Kbvals, Kcvals, ansTab, TS,
vars, update)
local Ka,Kb,Kc, st,et, TSX,TSY,TSZ, i, N,var,Lrt,sys, zero,
zeroInd,posCell, numRoots, d, cad, rcp, rcpolys, rcinequal,
count, R, newR, sol:

R := RegularChains:-PolynomialRing(vars):

count := 0:

for Ka in Kavals do
for Kb in Kbvals do
for Kc in Kcvals do

    st := time():

    # Substitutions
    TSZ := subs(k28=Ka, k29=Kb, k30=Kc, TS):

    # Lazy Real Triangularize and checks
    Lrt := LazyRealTriangularize(TSZ, R):
    if type(Lrt,list)<>true then
      error("Lrt output format changed (not list) - probably
something lazy.  At: ", [Ka, Kb, Kc]):
    fi:
    N := nops( Lrt ):
    if N<>1 then
      error("Multiple solution components found - unexpected.
At: ", [Ka, Kb, Kc] ): fi:
    sys := Lrt[1]:
    rcpolys := map(X->lhs(X), select(X->op(0,X)=`=`, sys)):
    rcinequal := map(X->rhs(X), remove(X->op(0,X)=`=`, sys)):
    N := nops(rcpolys):
    if N<>nops(vars) then
      error("Solution component has unexpected number of
equations.  At: ", [Ka, Kb, Kc] ):
    fi:
    if (rcinequal<>[vars[-1]] and rcinequal<>[vars[-1], vars
[-2]]) then
```

```
      error("Solution component has unexpected inequality.  At:
", [Ka, Kb, Kc] ):
    fi:
    for i from 1 to nops(vars)-1 do:
      d := degree(rcpolys[i], vars[i]):
      if d <>1 then
        error("polynomials are not linear - unexpected.  At: ",
[Ka, Kb, Kc] ):
      fi:
    od:
    d := degree(rcpolys[nops(vars)], vars[-1]):
    if d <>8 then
      error("Final polynomial is not degree 8 - unexpected.
At: ", [Ka, Kb, Kc] ):
    fi:

    # Creating CAD and counting
    newR := PolynomialRing([vars[-2], vars[-1]]):
    cad := CylindricalAlgebraicDecompose( [rcpolys[-1], rcpolys
[-2], vars[-1], vars[-2]], newR, output=cadcell ):
    sol := select(X-> SignAtBox( rcpolys[-2], X[SamplePoint],
newR)=0, cad);
    sol := select(X-> SignAtBox( rcpolys[-1], X[SamplePoint],
newR)=0, sol);
    sol := select(X-> SignAtBox( vars[-1], X[SamplePoint],
newR)=1, sol);
    sol := select(X-> SignAtBox( vars[-2], X[SamplePoint],
newR)=1, sol);
    numRoots := nops(sol):

    if update = false then
      ;
    else
      count := count + 1:
      if modp(count,update)=0 then
          print("so far...", count):
      fi:
    fi:

    et := time() - st:
    ansTab[Ka, Kb, Kc] := [numRoots, et]:


od: od: od:

end proc:


generate_innerCO := proc(Kavals, Kbvals, Kcvals, ansTab, TS,
vars, update)
local Ka,Kb,Kc, st,et, TSX,TSY,TSZ, i, N,var,Lrt,sys, zero,
zeroInd,posCell, numRoots, d, cad, rcp, rcpolys, rcinequal,
count, R, newR, sol, TSZpols:

R := RegularChains:-PolynomialRing(vars):

count := 0:
```

```
for Ka in Kavals do
for Kb in Kbvals do
for Kc in Kcvals do

    st := time():

    # Substitutions
    TSZ := subs(k28=Ka, k29=Kb, k30=Kc, TS):

    TSZpols := map(X->lhs(X), select(X->op(0,X)=`=`, TSZ)):

    # Creating CAD and counting
    newR := PolynomialRing([vars[-2], vars[-1]]):
    cad := CylindricalAlgebraicDecompose( [op(TSZpols), vars
[-1], vars[-2]], newR, output=cadcell ):
    sol := select(X-> SignAtBox( TSZpols[-2], X[SamplePoint],
newR)=0, cad);
    sol := select(X-> SignAtBox( TSZpols[-1], X[SamplePoint],
newR)=0, sol);
    sol := select(X-> SignAtBox( vars[-1], X[SamplePoint],
newR)=1, sol);
    sol := select(X-> SignAtBox( vars[-2], X[SamplePoint],
newR)=1, sol);
    numRoots := nops(sol):

    if update = false then
      ;
    else
      count := count + 1:
      if modp(count,update)=0 then
          print("so far...", count):
      fi:
    fi:

    et := time() - st:
    ansTab[Ka, Kb, Kc] := [numRoots, et]:

od: od: od:

end proc:


generate_howMany := proc(KaR, KbR, KcR, ansTab)
local check, r, Kavals, Kbvals, Kcvals, ans, i, count:

for r in [KaR, KbR, KcR] do
  check := (type(r, list) and nops(r)=3 and r[2]>r[1]) or type
(r, integer):
  if check<>true then
    error("First 3 arguments must each be an integer or a list
of 3 integers (start, stop, jump)"):
  fi:
od:

if type(KaR, integer) then
    Kavals:= [KaR]:
else
```

```
    Kavals := []:
    for i from KaR[1] by KaR[3] to KaR[2] do
      Kavals := [op(Kavals), i]:
    od:
fi:

if type(KbR, integer) then
    Kbvals:= [KbR]:
else
    Kbvals := []:
    for i from KbR[1] by KbR[3] to KbR[2] do
      Kbvals := [op(Kbvals), i]:
    od:
fi:

if type(KcR, integer) then
    Kcvals:= [KcR]:
else
    Kcvals := []:
    for i from KcR[1] by KcR[3] to KcR[2] do
      Kcvals := [op(Kcvals), i]:
    od:
fi:

print([nops(Kavals),nops(Kbvals),nops(Kcvals)]);
return( nops(Kavals)*nops(Kbvals)*nops(Kcvals) ):

end proc:
```

The problem = MAPK 028

```
> TS := [
  k2*x9 + k8*x10 + k21*x15 + k26*x16 - x1*x5*(k1 + k7) - x1*x6*
  (k22 + k27),
  k3*x9 + k5*x7 + k24*x12 - k4*x2*x5 - k23*x2*x6,
  k9*x10 + k11*x8 + k16*x13 + k19*x14 - x3*x6*(k17 + k18) - k10*
  x3*x5,
  k6*x7 + k12*x8 + k14*x11 - k13*x4*x6,
  x9*(k2 + k3) + x7*(k5 + k6) + x10*(k8 + k9) + x8*(k11 + k12) -
  x1*x5*(k1 + k7) - k4*x2*x5 - k10*x3*x5,
  k14*x11 + k16*x13 + k19*x14 + k21*x15 + k24*x12 + k26*x16 - x3*
  x6*(k17 + k18) - x1*x6*(k22 + k27) - k13*x4*x6 - k23*x2*x6,
  k4*x2*x5 - x7*(k5 + k6),
  k10*x3*x5 - x8*(k11 + k12),
  k1*x1*x5 - x9*(k2 + k3),
  k7*x1*x5 - x10*(k8 + k9),
  k13*x4*x6 - x11*(k14 + k15),
  k23*x2*x6 - x12*(k24 + k25),
  k15*x11 - k16*x13 + k17*x3*x6,
  k18*x3*x6 - x14*(k19 + k20),
  k20*x14 - k21*x15 + k22*x1*x6,
  k25*x12 - k26*x16 + k27*x1*x6,
  x6 - k28 + x11 + x12 + x13 + x14 + x15 + x16,
  x5 - k29 + x7 + x8 + x9 + x10,
  x1 - k30 + x2 + x3 + x4 + x7 + x8 + x9 + x10 + x11 + x12 + x13
  + x14 + x15 + x16
  ];
```

$$TS := [k2\,x9 + k8\,x10 + k21\,x15 + k26\,x16 - x1\,x5\,(k1 + k7) - x1\,x6\,(k22 + k27),\ -k23\,x2\,x6 - k4\,x2\,x5 + k24\,x12 + k3\,x9 + k5\,x7,\ k9\,x10 + k11\,x8 + k16\,x13 + k19\,x14 - x3\,x6\,(k17 + k18) - k10\,x3\,x5,\ -k13\,x4\,x6 + k12\,x8 + k14\,x11 + k6\,x7,\ x9\,(k2 + k3) + x7\,(k5 + k6) + x10\,(k8 + k9) + x8\,(k11 + k12) - x1\,x5\,(k1 + k7) - k4\,x2\,x5 - k10\,x3\,x5,\ k14\,x11 + k16\,x13 + k19\,x14 + k21\,x15 + k24\,x12 + k26\,x16 - x3\,x6\,(k17 + k18) - x1\,x6\,(k22 + k27) - k13\,x4\,x6 - k23\,x2\,x6,\ k4\,x2\,x5 - x7\,(k5 + k6),\ k10\,x3\,x5 - x8\,(k11 + k12),\ k1\,x1\,x5 - x9\,(k2 + k3),\ k7\,x1\,x5 - x10\,(k8 + k9),\ k13\,x4\,x6 - x11\,(k14 + k15),\ k23\,x2\,x6 - x12\,(k24 + k25),\ k17\,x3\,x6 + k15\,x11 - k16\,x13,\ k18\,x3\,x6 - x14\,(k19 + k20),\ k22\,x1\,x6 + k20\,x14 - k21\,x15,\ k27\,x1\,x6 + k25\,x12 - k26\,x16,\ x6 - k28 + x11 + x12 + x13 + x14 + x15 + x16,\ x5 - k29 + x7 + x8 + x9 + x10,\ x1 - k30 + x2 + x3 + x4 + x7 + x8 + x9 + x10 + x11 + x12 + x13 + x14 + x15 + x16]$$

```
> nops(TS);
```

19

```
> indets(TS);
```

$$\{k1, k10, k11, k12, k13, k14, k15, k16, k17, k18, k19, k2, k20, k21, k22, k23, k24, k25, k26, k27, k28, k29, k3, k30, k4, k5, k6, k7, k8, k9, x1, x10, x11, x12, x13, x14, x15, x16, x2, x3, x4, x5, x6, x7, x8, x9\}$$

```
> s := [
  k1 = 0.005,
  k2 = 1,
  k3 = 1.08,
```

```
  k4 = 0.025,
  k5 = 1,
  k6 = 0.007,
  k7 = 0.05,
  k8 = 1,
  k9 = 0.008,
  k10 = 0.005,
  k11 = 1,
  k12 = 0.45,
  k13 = 0.045,
  k14 = 1,
  k15 = 0.092,
  k16 = 1,
  k17 = 0.01,
  k18 = 0.01,
  k19 = 1,
  k20 = 0.5,
  k21 = 0.086,
  k22 = 0.0011,
  k23 = 0.01,
  k24 = 1,
  k25 = 0.47,
  k26 = 0.14,
  k27 = 0.0018];
```

$$s := [k1 = 0.005, k2 = 1, k3 = 1.08, k4 = 0.025, k5 = 1, k6 = 0.007, k7 = 0.05, k8 = 1, k9 = 0.008, k10 = 0.005, k11 = 1, k12 = 0.45, k13 = 0.045, k14 = 1, k15 = 0.092, k16 = 1, k17 = 0.01, k18 = 0.01, k19 = 1, k20 = 0.5, k21 = 0.086, k22 = 0.0011, k23 = 0.01, k24 = 1, k25 = 0.47, k26 = 0.14, k27 = 0.0018]$$

```
> subs(s,TS); indets(%);
```

$$[x9 + x10 + 0.086\,x15 + 0.14\,x16 - 0.055\,x1\,x5 - 0.0029\,x1\,x6,\ -0.025\,x2\,x5 - 0.01\,x2\,x6 + x12 + x7 + 1.08\,x9,\ 0.008\,x10 + x8 + x13 + x14 - 0.02\,x3\,x6 - 0.005\,x3\,x5,\ -0.045\,x4\,x6 + x11 + 0.007\,x7 + 0.45\,x8,\ 2.08\,x9 + 1.007\,x7 + 1.008\,x10 + 1.45\,x8 - 0.055\,x1\,x5 - 0.025\,x2\,x5 - 0.005\,x3\,x5,\ x11 + x13 + x14 + 0.086\,x15 + x12 + 0.14\,x16 - 0.02\,x3\,x6 - 0.0029\,x1\,x6 - 0.045\,x4\,x6 - 0.01\,x2\,x6,\ 0.025\,x2\,x5 - 1.007\,x7,\ 0.005\,x3\,x5 - 1.45\,x8,\ 0.005\,x1\,x5 - 2.08\,x9,\ 0.05\,x1\,x5 - 1.008\,x10,\ 0.045\,x4\,x6 - 1.092\,x11,\ 0.01\,x2\,x6 - 1.47\,x12,\ 0.01\,x3\,x6 + 0.092\,x11 - x13,\ 0.01\,x3\,x6 - 1.5\,x14,\ 0.0011\,x1\,x6 + 0.5\,x14 - 0.086\,x15,\ 0.0018\,x1\,x6 + 0.47\,x12 - 0.14\,x16,\ x6 - k28 + x11 + x12 + x13 + x14 + x15 + x16,\ x5 - k29 + x7 + x8 + x9 + x10,\ x1 - k30 + x2 + x3 + x4 + x7 + x8 + x9 + x10 + x11 + x12 + x13 + x14 + x15 + x16]$$

$$\{k28, k29, k30, x1, x10, x11, x12, x13, x14, x15, x16, x2, x3, x4, x5, x6, x7, x8, x9\}$$

```
> vars := [x16,x15,x14,x13,x12,x11,x10,x9,x8,x7,x6,x5,x4,x3,x2,
  x1]: R:=PolynomialRing(vars):
> TSS := subs(s,TS):
  convert(TSS, rational): TSY := map(X->X*10000, %):
  TSX := TSY:
  for entry in vars do: TSX := [op(TSX), entry>0]: od:
  TSX;
```

$$[-550\,x1\,x5 - 29\,x1\,x6 + 10000\,x10 + 860\,x15 + 1400\,x16 + 10000\,x9,\ -250\,x2\,x5 - 100\,x2\,x6 + 10000\,x12 + 10000\,x7 + 10800\,x9,\ -50\,x3\,x5 - 200\,x3\,x6 + 80\,x10 + 10000\,x13 + 10000\,x14 + 10000\,x8,\ -450\,x4\,x6 + 10000\,x11 + 70\,x7 + 4500\,x8,\ -550\,x1\,x5$$

$$-250\,x2\,x5 - 50\,x3\,x5 + 10080\,x10 + 10070\,x7 + 14500\,x8 + 20800\,x9,\ -29\,x1\,x6 - 100\,x2\,x6 - 200\,x3\,x6 - 450\,x4\,x6 + 10000\,x11 + 10000\,x12 + 10000\,x13 + 10000\,x14 + 860\,x15 + 1400\,x16,\ 250\,x2\,x5 - 10070\,x7,\ 50\,x3\,x5 - 14500\,x8,\ 50\,x1\,x5 - 20800\,x9,\ 500\,x1\,x5 - 10080\,x10,\ 450\,x4\,x6 - 10920\,x11,\ 100\,x2\,x6 - 14700\,x12,\ 100\,x3\,x6 + 920\,x11 - 10000\,x13,\ 100\,x3\,x6 - 15000\,x14,\ 11\,x1\,x6 + 5000\,x14 - 860\,x15,\ 18\,x1\,x6 + 4700\,x12 - 1400\,x16,\ 10000\,x6 - 10000\,k28 + 10000\,x11 + 10000\,x12 + 10000\,x13 + 10000\,x14 + 10000\,x15 + 10000\,x16,\ 10000\,x5 - 10000\,k29 + 10000\,x7 + 10000\,x8 + 10000\,x9 + 10000\,x10,\ 10000\,x1 - 10000\,k30 + 10000\,x2 + 10000\,x3 + 10000\,x4 + 10000\,x7 + 10000\,x8 + 10000\,x9 + 10000\,x10 + 10000\,x11 + 10000\,x12 + 10000\,x13 + 10000\,x14 + 10000\,x15 + 10000\,x16,\ 0 < x16,\ 0 < x15,\ 0 < x14,\ 0 < x13,\ 0 < x12,\ 0 < x11,\ 0 < x10,\ 0 < x9,\ 0 < x8,\ 0 < x7,\ 0 < x6,\ 0 < x5,\ 0 < x4,\ 0 < x3,\ 0 < x2,\ 0 < x1\ ]$$

Graphs and Timings - Original

```
> generate_howMany( [40,160,20], 180, [100,1600,200]);
```

[7, 1, 8]

56

```
> answersL:=table():
  generate( [40,160,20], 180, [100,1600,200], answersL, TSX,
  vars, 10):
```

"so far...", 10

"so far...", 20

"so far...", 30

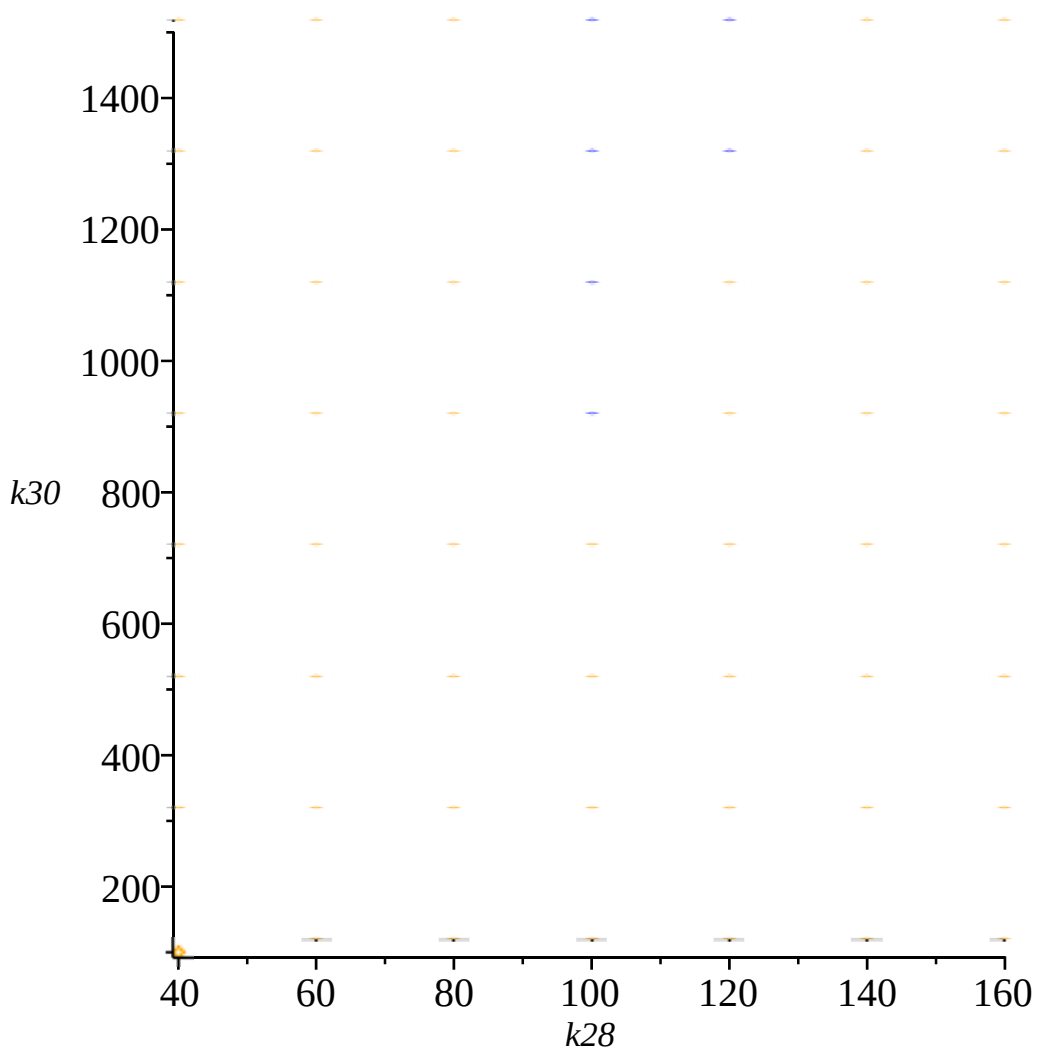
"so far...", 40

"so far...", 50

```
> save(answersL, "SamplePoints-Sys28-Original-Left.txt"):
> read("SamplePoints-Sys28-Original-Left.txt"):
> inds := [indices( answersL )]: nops(%);
  ents := [entries( answersL )]: nops(%):
  nums := map(X->X[1][1], ents):
  timeL := map(X->X[1][2], ents):
  cols := []:
  for ent in nums do:
  if ent=1 then
    cols := [op(cols), "Orange"]:
  elif ent=3 then:
    cols := [op(cols), "Blue"]:
  else
    cols := [op(cols), "Red"]:
  fi:
  od:
```

56

```
> plots :- pointplot( map(X->[X[1],X[3]],inds), color = cols,
  labels=[k28, k30] );
```




```
> Statistics:-Mean(timeL);
```

44.3160000000000

```
> max(timeL);
```

84.116

```
> Statistics:-Median(timeL);
```

41.6130000000000

```
> Statistics:-StandardDeviation(timeL);
```

10.3982611920543

```
> generate_howMany( 100, [120,240,20], [100,1600,200]);
```

[1, 7, 8]

56

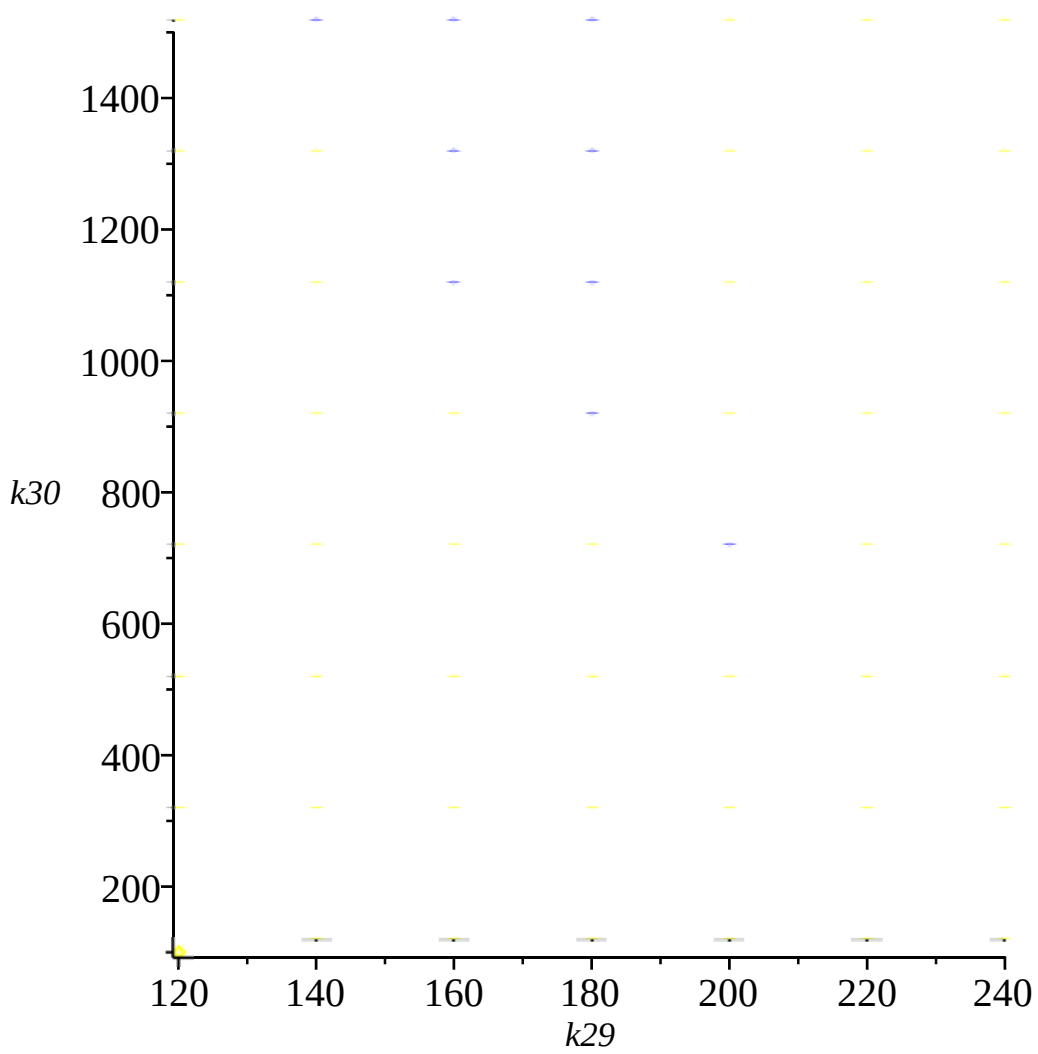
```
> answersR:=table():
  generate( 100, [120,240,20], [100,1600,200], answersR, TSX,
  vars, 10):
  save(answersR, "SamplePoints-Sys28-Original-Right.txt"):
```

"so far...", 10

```
                                "so far...", 20
                                "so far...", 30
                                "so far...", 40
                                "so far...", 50
> save(answersR, "SamplePoints-Sys28-Original-Right.txt"):
> read("SamplePoints-Sys28-Original-Right.txt"):
> inds := [indices( answersR )]: nops(%);
  ents := [entries( answersR )]: nops(%):
  nums := map(X->X[1][1], ents):
  timeR := map(X->X[1][2], ents):
  cols := []:
  for ent in nums do:
  if ent=1 then
    cols := [op(cols), "Yellow"]:
  elif ent=3 then:
    cols := [op(cols), "Blue"]:
  else
    cols := [op(cols), "Red"]:
  fi:
  od:
```



```
> plots :- pointplot( map(X->[X[2],X[3]],inds), color = cols,
  labels=[k29, k30] );
```




```
> Statistics:-Mean(timeR);
```

40.5435535714286

```
> timeAll := [op(timeR), op(timeL)]:
> Statistics:-Mean(timeAll);
```

42.4297767857143

```
> Statistics:-Median(timeAll);
```

40.5290000000000

```
> Statistics:-StandardDeviation(timeAll);
```

8.63227648584688

```
> max(timeAll);
```

84.116

Reduced Problem

```
> TS := [
    3796549898085*k29*x5^3*x6 + 71063292573000*k29*x5^3 +
  106615407090630*k29*x5^2*x6^2 + 479383905861000*k29*x5^2*x6 +
  299076127852260*k29*x5*x6^3 + 3505609439955600*k29*x5*x6^2 +
  91244417457024*k29*x6^4 + 3557586742819200*k29*x6^3 -
  598701732300*k30*x5^3*x6 - 83232870778950*k30*x5^2*x6^2 -
  185019487578700*k30*x5*x6^3 - 3796549898085*x5^4*x6 -
  71063292573000*x5^4 - 106615407090630*x5^3*x6^2 -
  479383905861000*x5^3*x6 - 299076127852260*x5^2*x6^3 -
  3505609439955600*x5^2*x6^2 - 91244417457024*x5*x6^4 -
  3557586742819200*x5*x6^3 = 0,
    3796549898085*k28*x5^3*x6 + 71063292573000*k28*x5^3 +
  106615407090630*k28*x5^2*x6^2 + 479383905861000*k28*x5^2*x6 +
  299076127852260*k28*x5*x6^3 + 3505609439955600*k28*x5*x6^2 +
  91244417457024*k28*x6^4 + 3557586742819200*k28*x6^3 -
  3197848165785*k30*x5^3*x6 - 23382536311680*k30*x5^2*x6^2 -
  1140566402735 60*k30*x5*x6^3 - 91244417457024*k30*x6^4 -
  3796549898085*x5^3*x6^2 - 71063292573000*x5^3*x6 -
  106615407090630*x5^2*x6^3 - 479383905861000*x5^2*x6^2 -
  299076127852260*x5*x6^4 - 3505609439955600*x5*x6^3 -
  91244417457024*x6^5 - 3557586742819200*x6^4 = 0];
```

$$TS := [3796549898085\,k29\,x5^3\,x6 + 106615407090630\,k29\,x5^2\,x6^2 + 299076127852260\,k29\,x5\,x6^3 + 91244417457024\,k29\,x6^4 - 598701732300\,k30\,x5^3\,x6 - 83232870778950\,k30\,x5^2\,x6^2 - 185019487578700\,k30\,x5\,x6^3 - 3796549898085\,x5^4\,x6 - 106615407090630\,x5^3\,x6^2 - 299076127852260\,x5^2\,x6^3 - 91244417457024\,x5\,x6^4 + 71063292573000\,k29\,x5^3 + 479383905861000\,k29\,x5^2\,x6 + 3505609439955600\,k29\,x5\,x6^2 + 3557586742819200\,k29\,x6^3 - 71063292573000\,x5^4 - 479383905861000\,x5^3\,x6 - 3505609439955600\,x5^2\,x6^2 - 3557586742819200\,x5\,x6^3 = 0,\ 3796549898085\,k28\,x5^3\,x6 + 106615407090630\,k28\,x5^2\,x6^2 + 299076127852260\,k28\,x5\,x6^3 + 91244417457024\,k28\,x6^4 - 3197848165785\,k30\,x5^3\,x6 - 23382536311680\,k30\,x5^2\,x6^2 - 114056640273560\,k30\,x5\,x6^3 - 91244417457024\,k30\,x6^4 - 3796549898085\,x5^3\,x6^2 - 106615407090630\,x5^2\,x6^3 - 299076127852260\,x5\,x6^4 - 91244417457024\,x6^5 + 71063292573000\,k28\,x5^3 + 479383905861000\,k28\,x5^2\,x6 + 3505609439955600\,k28\,x5\,x6^2 + 3557586742819200\,k28\,x6^3 - 71063292573000\,x5^3\,x6 - 479383905861000\,x5^2\,x6^2 - 3505609439955600\,x5\,x6^3 - 3557586742819200\,x6^4 = 0]$$

```
> TSX := [op(TS), x6>0, x5>0]:
> vars := [x6,x5]: R:=PolynomialRing(vars):
```

Graphs and Timings - Reduced

```
> generate_howMany( [40,160,10], 180, [100,1600,100]);
```

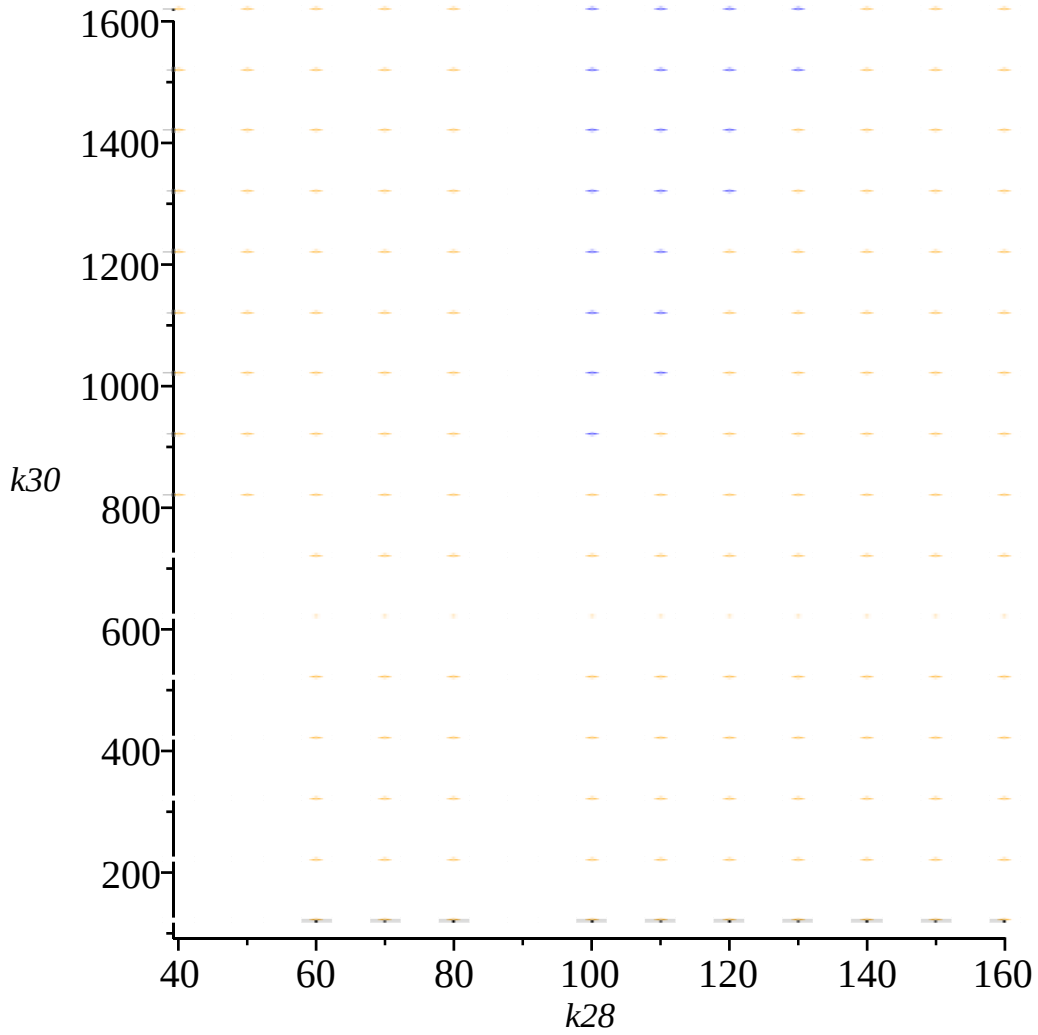
[13, 1, 16]

208

```
> answersL:=table():
  generate( [40,160,10], 180, [100,1600,100], answersL, TSX,
  vars):
> save(answersL, "SamplePoints-Sys28-Reduced-Left.txt"):
> read("SamplePoints-Sys28-Reduced-Left.txt"):
> inds := [indices( answersL )]: nops(%);
  ents := [entries( answersL )]: nops(%):
  nums := map(X->X[1][1], ents):
  timeL := map(X->X[1][2], ents):
  cols := []:
  for ent in nums do:
  if ent=1 then
    cols := [op(cols), "Orange"]:
  elif ent=3 then:
    cols := [op(cols), "Blue"]:
  else
    cols := [op(cols), "Red"]:
  fi:
  od:
```

208

```
> plots :- pointplot( map(X->[X[1],X[3]],inds), color = cols,
  labels=[k28, k30] );
```




```
> Statistics:-Mean(timeL);
```

0.470403846153846

```
> generate_howMany( 100, [120,240,10], [100,1600,100]);
```

[1, 13, 16]

208

```
> answersR:=table():
  generate( 100, [120,240,10], [100,1600,100], answersR, TSX,
  vars):
> save(answersR, "SamplePoints-Sys28-Reduced-Right.txt"):
> read("SamplePoints-Sys28-Reduced-Right.txt"):
> inds := [indices( answersR )]: nops(%);
  ents := [entries( answersR )]: nops(%):
  nums := map(X->X[1][1], ents):
  timeR := map(X->X[1][2], ents):
  cols := []:
  for ent in nums do:
  if ent=1 then
    cols := [op(cols), "Yellow"]:
```