```
> restart; kernelopts(opaquemodules=false):
```

We are using the standard Maple 18 Library

```
> libname:="/home/me350/Programs/Maple18/lib";
```

$$libname := \text{"/home/me350/Programs/Maple18/lib"} \quad (1)$$

We make use of the following packages:

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

Additionally, we use Maple code written at the University of Bath: The ProjectionCAD package (should be hosted alongside this worksheet).

```
> read("ProjectionCAD.mpl"):
  with(ProjectionCAD):
```

"This is V3.18 of the ProjectionCAD module from 11th February 2015, designed and tested for use in Maple 18." (2)

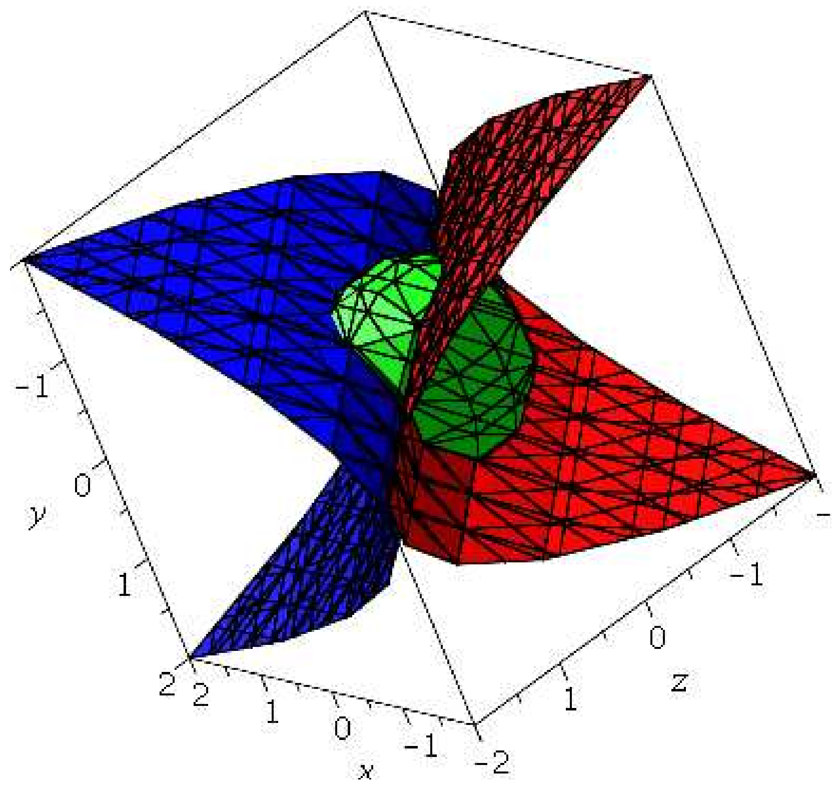
Example 1

```
> f1 := x+y^2+z:
  f2 := x-y^2+z:
  g := x^2+y^2+z^2-1:
  F := [f1,g,f2]:
  lsas := [[f1=0, f2=0, g>=0]];
  ord := [z,y,x]:
  R := PolynomialRing(ord):
```

$$lsas := [[y^2 + x + z = 0, -y^2 + x + z = 0, 0 \le x^2 + y^2 + z^2 - 1]] \quad (1.1)$$

Plot

```
> implicitplot3d([f1,f2,g], x=-2..2, y=-2..2, z=-2..2, color=
  [red,blue,green]);
```



-1
0
y
1
2
2
1
0
-1
-2
x
1
0
z
-1

Solution by hand

For f1 and f2 to be satisfied together we need their resultant to be zero:

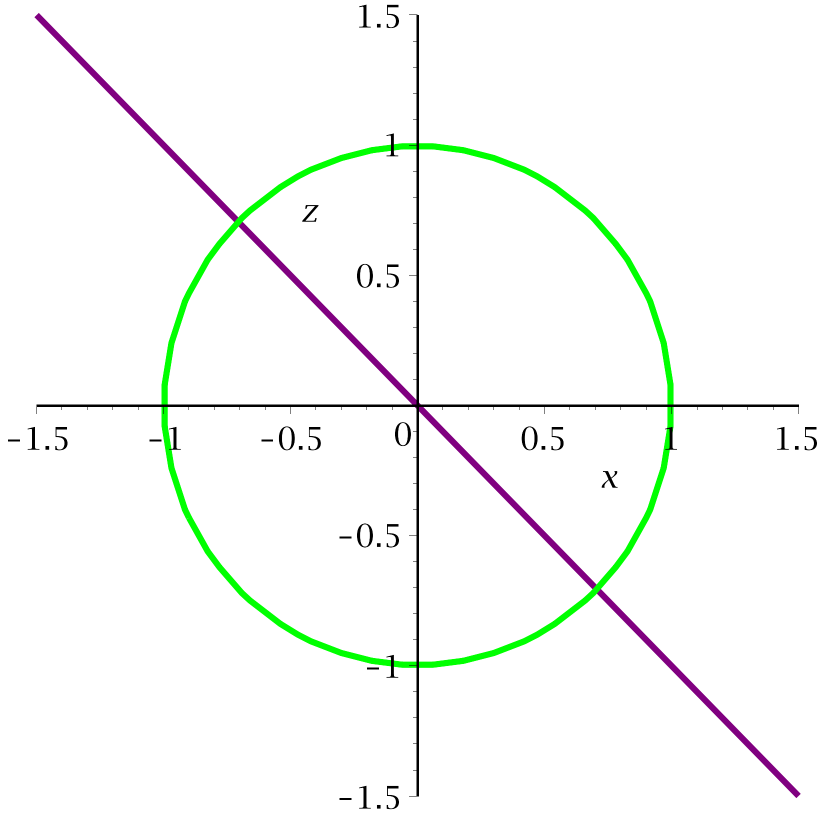
```
> solve(f1=f2);
  resultant(f1,f2,z);
```

$$\{x = x, y = 0, z = z\}$$

$$-2\,y^2 \qquad \textbf{(1.1.1)}$$

```
> subs(y=0,[f1=0,g]);
  plots:-implicitplot(%, x=-1.5..1.5, z=-2..2, color=
  ["Purple",green], thickness=3);
```

$$[x + z = 0,\ x^2 + z^2 - 1]$$



So the solution set is all z=-x, y=0 and |x|>sqrt(2)/2.

Sign-invariant CAD

Using the CAD from the RegularChains Library

```
> RCcad := SemiAlgebraicSetTools:-
  CylindricalAlgebraicDecompose(F, R, output=list): nops(%);
                                   1487                                  (1.2.1)
```

Using our own ProjectionCAD

```
> ProjectionCAD:-CADFull(F, ord, method=McCallum, output=list)
  : nops(%);
                                   1487                                  (1.2.2)
```

Qepcad also produced 1487 cells.

One equational constraint and lifting with EC only

Using ProjectionCAD and the declaration of one EC:

```
> cad := ProjectionCAD:-ECCAD( [f1,[f2,g]], ord, output=
  listwithrep): nops(%);
  ProjectionCAD:-ECCAD( [f2,[f1,g]], ord, output=list): nops
  (%);
                                    141
                                    141                                  (1.3.1)
```

Note that this algorithm lifts only with respect to the declared EC at the final stage. This is why the cell count is lower than Qepcad (289).

Biequational projection and lifting with ECs only

Now we make use of both ECs in the formula. We do this by using the implicit EC at the second level:

```
> RR:=resultant(f1, f2, z);
```

$$RR := -2\,y^2 \tag{1.4.1}$$

Hence we define

```
> A:={f1,f2,g};
  E:={f1};
```

$$A := \{-y^2+x+z,\ y^2+x+z,\ x^2+y^2+z^2-1\}$$
$$E := \{y^2+x+z\} \tag{1.4.2}$$

and then A' and E' are

```
> Ad := [op(ProjectionCAD:-ECCADProjOp( E, A, [z,y,x])) ];
  Ed := [RR];
```

$$Ad := [-2\,y^2,\ y^4+2\,x\,y^2+2\,x^2+y^2-1]$$
$$Ed := [-2\,y^2] \tag{1.4.3}$$

Finally the univariate projection polynomials are:

```
> ProjectionCAD:-ECCADProjOp({y^2}, convert(Ad,set), [y,x]);
```

$$\{(2\,x^2-1)^2\} \tag{1.4.4}$$

So the CAD of the real line has 5 cells

```
> ProjectionCAD:-CADFull( [2*x^2-1], [x], output=piecewise,
  method=McCallum);
```

$$\begin{cases} [\mathit{regular\_chain},\ [[-2,\ -2]]] & x < -\frac{\sqrt{2}}{2} \\ \left[\mathit{regular\_chain},\ \left[\left[-\frac{91}{128},\ -\frac{45}{64}\right]\right]\right] & x = -\frac{\sqrt{2}}{2} \\ [\mathit{regular\_chain},\ [[0,\ 0]]] & -\frac{\sqrt{2}}{2} < x < \frac{\sqrt{2}}{2} \\ \left[\mathit{regular\_chain},\ \left[\left[\frac{45}{64},\ \frac{91}{128}\right]\right]\right] & x = \frac{\sqrt{2}}{2} \\ [\mathit{regular\_chain},\ [[2,\ 2]]] & \frac{\sqrt{2}}{2} < x \end{cases} \tag{1.4.5}$$

```
> cadR1 := ProjectionCAD:-CADFull( [2*x^2-1], [x], output=
  listwithrep, method=McCallum): nops(%);
```

$$5 \qquad (1.4.6)$$

The cad of the plane is found by lifting with R only. I.e. same as ECCAD command on E' and A'

```
> ProjectionCAD:-ECCAD( [op(Ed), Ad], [y,x], output=piecewise
  );
  cadR2:=ProjectionCAD:-ECCAD( [op(Ed), Ad], [y,x], output=
  listwithrep ): nops(%);
```

$$\begin{cases} \begin{cases} [regular\_chain, [[-2, -2], [-1, -1]]] & y < 0 \\ [regular\_chain, [[-2, -2], [0, 0]]] & y = 0 \\ [regular\_chain, [[-2, -2], [1, 1]]] & 0 < y \end{cases} & x < -\frac{\sqrt{2}}{2} \\ \begin{cases} \left[regular\_chain, \left[\left[-\frac{91}{128}, -\frac{45}{64}\right], [-1, -1]\right]\right] & y < 0 \\ \left[regular\_chain, \left[\left[-\frac{91}{128}, -\frac{45}{64}\right], [0, 0]\right]\right] & y = 0 \\ \left[regular\_chain, \left[\left[-\frac{91}{128}, -\frac{45}{64}\right], [1, 1]\right]\right] & 0 < y \end{cases} & x = -\frac{\sqrt{2}}{2} \\ \begin{cases} [regular\_chain, [[0, 0], [-1, -1]]] & y < 0 \\ [regular\_chain, [[0, 0], [0, 0]]] & y = 0 \\ [regular\_chain, [[0, 0], [1, 1]]] & 0 < y \end{cases} & -\frac{\sqrt{2}}{2} < x < \frac{\sqrt{2}}{2} \\ \begin{cases} \left[regular\_chain, \left[\left[\frac{45}{64}, \frac{91}{128}\right], [-1, -1]\right]\right] & y < 0 \\ \left[regular\_chain, \left[\left[\frac{45}{64}, \frac{91}{128}\right], [0, 0]\right]\right] & y = 0 \\ \left[regular\_chain, \left[\left[\frac{45}{64}, \frac{91}{128}\right], [1, 1]\right]\right] & 0 < y \end{cases} & x = \frac{\sqrt{2}}{2} \\ \begin{cases} [regular\_chain, [[2, 2], [-1, -1]]] & y < 0 \\ [regular\_chain, [[2, 2], [0, 0]]] & y = 0 \\ [regular\_chain, [[2, 2], [1, 1]]] & 0 < y \end{cases} & \frac{\sqrt{2}}{2} < x \end{cases}$$

$$15 \qquad (1.4.7)$$

Finally, the CAD of R^3 is found by lifting each cell in the plane with respect to one of the ECs.   In this example, we have the same cell count with either.

```
> out := []:
  for i from 1 to nops(cadR2) do
    stk := ProjectionCAD:-CADGenerateStack( cadR2[i], map
  (expand,[f1]), [z,y,x], output=listwithrep):
    out := [op(out), op(stk)]:
  od:
  nops(out);
```

$$45 \tag{1.4.8}$$

```
> #ProjectionCAD:-PCAD_LWRCADtoPWCAD(out);
```

```
> out := []:
  for i from 1 to nops(cadR2) do
    stk := ProjectionCAD:-CADGenerateStack( cadR2[i], map
  (expand,[f2]), [z,y,x], output=listwithrep):
    out := [op(out), op(stk)]:
  od:
  nops(out);
```

$$45 \tag{1.4.9}$$

Biequational projection, lifting with ECs only, lifting over EC sections only

We start and proceed to the CAD of the plane as before

```
> RR:=resultant(f1, f2, z):
```

```
> A:={f1,f2,g};
  E:={f1};
```

$$A := \{-y^2+z+x,\ y^2+z+x,\ x^2+y^2+z^2-1\}$$

$$E := \{y^2+z+x\} \qquad \textbf{(1.5.1)}$$

```
> Ad := [ op(ProjectionCAD:-ECCADProjOp( E, A, [z,y,x])) ];
  Ed := [RR];
```

$$Ad := [-2\,y^2,\ y^4+2\,x\,y^2+2\,x^2+y^2-1]$$

$$Ed := [-2\,y^2] \qquad \textbf{(1.5.2)}$$

```
> cadR2:=ProjectionCAD:-ECCAD( [op(Ed), Ad], [y,x], output=
  listwithrep ): nops(%);
```

$$15 \qquad \textbf{(1.5.3)}$$

Of these 21 cells, we only need worry about the cells where RR=0 (on all the others the formula is certainly false).
Since we liftied with RR only at the last step, we know that such cells are exactly those with last entry in the cell index even:

```
> select(X->X[1][-1]::even, cadR2); nops(%);
```

$$\left[\left[[1,2],\left[x<-\frac{\sqrt{2}}{2},\ y=0\right],[regular\_chain,[[-2,-2],[0,0]]]\right],\left[[2,2],\left[x=-\frac{\sqrt{2}}{2},\ y=0\right],\left[regular\_chain,\left[\left[-\frac{91}{128},-\frac{45}{64}\right],[0,0]\right]\right]\right],\left[[3,2],\left[-\frac{\sqrt{2}}{2}<x<\frac{\sqrt{2}}{2},\ y=0\right],[regular\_chain,[[0,0],[0,0]]]\right],\left[[4,2],\left[x=\frac{\sqrt{2}}{2},\ y=0\right],\left[regular\_chain,\left[\left[\frac{45}{64},\frac{91}{128}\right],[0,0]\right]\right]\right],\left[[5,2],\left[\frac{\sqrt{2}}{2}<x,\ y=0\right],[regular\_chain,[[2,2],[0,0]]]\right]\right]$$

$$5 \qquad \textbf{(1.5.4)}$$

For these cells we need to lift as before (with respect to an EC). For the others we only need to trivially extend to a cylinder (implemented as lifting with respect to a constant).

```
> out := []:
  for i from 1 to nops(cadR2) do
    cell := cadR2[i]:
    if cell[1][-1]::even then
      stk := ProjectionCAD:-CADGenerateStack( cadR2[i], map
  (expand,[f1]), [z,y,x], output=listwithrep):
    else
      stk := ProjectionCAD:-CADGenerateStack( cadR2[i], map
  (expand,[1]), [z,y,x], output=listwithrep):
    fi:
    out := [op(out), op(stk)]:
  od:
  nops(out);
                                   25                                  (1.5.5)
```

Hence we have a truth-invariant CAD of R^3 with only 25 cells.

```
> #ProjectionCAD:-PCAD_LWRCADtoPWCAD(out);
```