AmirHosein Sadeghimanesh
2021 November

This worksheet contains the Maple codes used for the simple example of the following paper.

AmirHosein Sadeghimanesh, Matthew England, Resultant Tools for Parametric Polynomial Systems with Application to Population Models, 2022.

# Simple quadratic equation

Consider the following polynomial in one variable 'x' and two parameters 'b' and 'c'.

```
> f := x^2 + b*x + c;
```

$$f := b\,x + x^2 + c \tag{1.1}$$

We are interested in partitioning the parameter plane to open connected sets such that the number of real solutions to f = 0 is invariant for the parameter choices coming from the same open set in the partition.
The Maple package 'RootFinding:-Parametric' can do this partitioning for us. However, here we want to find the discriminant variety manually ourselves using several different approaches explained in the paper.

Using Grobner basis computation.

```
> system1 := [f, diff(f,x)]; # adding f' because we want to know
  when two real solutions collide and become cmplex or the
  opposite.
  Grobner1 := Groebner:-Basis(system1, plex(x,c,b)); # computing
  the GB with a lexicographic order having x greater than b and
  c.
  basis1 := remove(has, Grobner1, [x]); # removing the
  polynomials containing x from Grobner1.
```

$$system1 := \left[b\,x + x^2 + c,\ b + 2\,x\right]$$

$$Grobner1 := \left[-b^2 + 4\,c,\ b + 2\,x\right]$$

$$basis1 := \left[-b^2 + 4\,c\right] \tag{1.2}$$

Therefore the curve seperating the regions where the number of real solutions may change, is defined by b^2-4*c.

```
> discriminantPart1 := basis1[1];
```

$$discriminantPart1 := -b^2 + 4\,c \tag{1.3}$$

Now if we also care about the sign of the real solutions, let say we want the number of positive real solutions being invariant, then in addition to the above curve, we also have another curve coming from the following computation. Instead of derivative of f, we have to consider x itself as the second polynomial.

```
> system2 := [f, x];
  Grobner2 := Groebner:-Basis(system2, plex(x,c,b));
  basis2 := remove(has, Grobner2, [x]);
```

$$system2 := [b\,x + x^2 + c, x]$$

$$Grobner2 := [c, x]$$

$$basis2 := [c] \tag{1.4}$$

```
> discriminantPart2 := basis2[1];
```

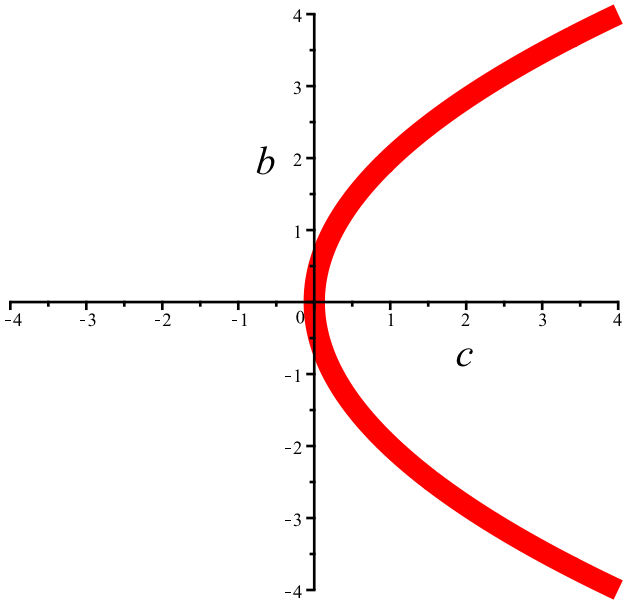
$$discriminantPart2 := c \tag{1.5}$$

The second curve is defined by c.

Plotting Figure 2.

Part a.

```
> discriminantCurve1 := plots:-implicitplot(discriminantPart1 =
  0, c=-4..4, b=-4..4, color=red, thickness=8, labels=[c,b],
  view=[-4..4,-4..4], labelfont=["NewTimesRoman",24]);
```

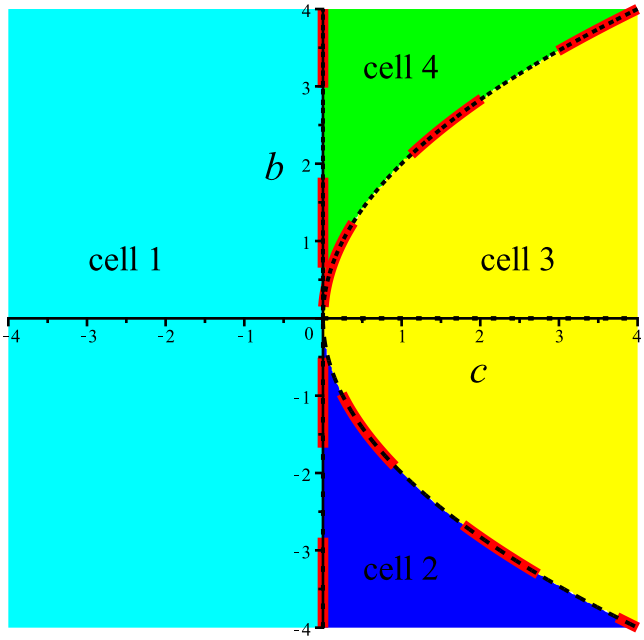



Part b.

Since the open CAD with respect to the above curve is trivial and easy to readily say, we are not

including some extra computation here. We just directly plot the final result.

```
> # curves
  pcurve1 := plots:-implicitplot(b^2-4*c = 0, c=-4..4, b=-4..4,
  color=red, thickness=4, linestyle=spacedash, labels=[c,b],
  view=[-4..4,-4..4], labelfont=["NewTimesRoman",24]):
  pcurve2 := plots:-implicitplot(c = 0, c=-4..4, b=-4..4, color=
  red, thickness=4, linestyle=spacedash, labels=[c,b], view=[-4.
  .4,-4..4], labelfont=["NewTimesRoman",24]):
  # regions
  pregion1 := plots:-inequal(c < 0, c=-4..4, b=-4..4, color=cyan)
  :
  pregion2 := plots:-inequal({c > 0, b < 0, b^2-4*c > 0}, c=-4.
  .4, b=-4..4, color=blue):
  pregion3 := plots:-inequal(b^2-4*c < 0, c=-4..4, b=-4..4,
  color=yellow):
  pregion4 := plots:-inequal({c > 0, b > 0, b^2-4*c > 0}, c=-4.
  .4, b=-4..4, color=green):
  # text
  ptext := plots:-textplot([[-3,0.5,"cell 1"], [0.5,-3.5,"cell
  2"], [2,0.5,"cell 3"], [0.5,3,"cell 4"]], align={above,right},
  font=["TimesNewRoman",20]):
  # fianl
  openCAD1 := plots:-display(pcurve1, pcurve2, pregion1,
  pregion2, pregion3, pregion4, ptext);
```

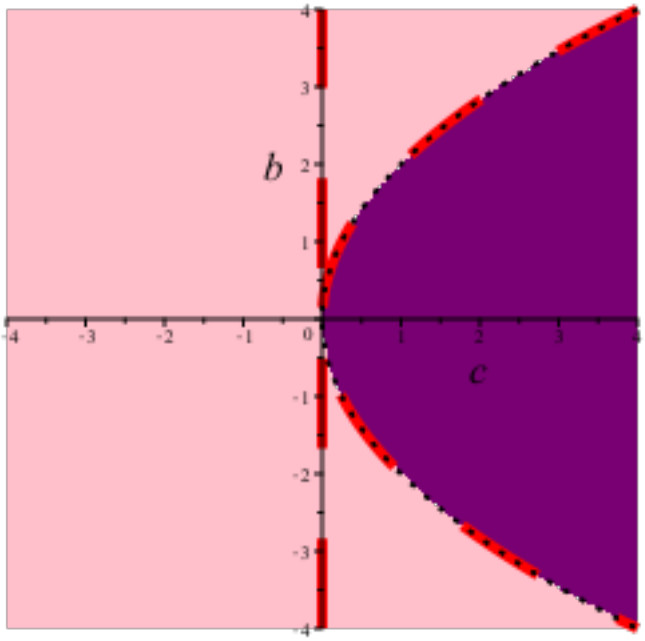



Part c.

Again since picking up a sample point from each of the above open cells and solving the system and counting the real roots can readily be done. Without putting any extra computation we just plot the final figure.

```
> # real solutions regions
  pregion5 := plots:-inequal(b^2-4*c > 0, c=-4..4, b=-4..4,
```

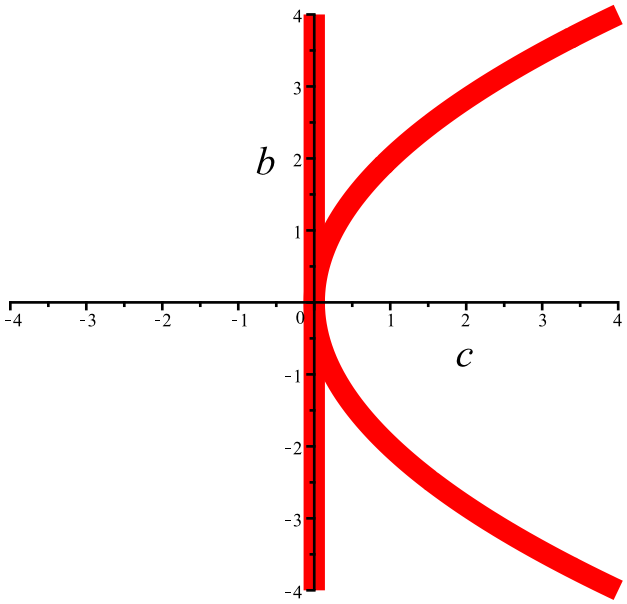
```
color=pink):
pregion6 := plots:-inequal(b^2-4*c < 0, c=-4..4, b=-4..4,
color=purple):
# final
fianlPartition1 := plots:-display(pcurve1, pcurve2, pregion5,
pregion6);
```




Plotting Figure 3.

Part a.

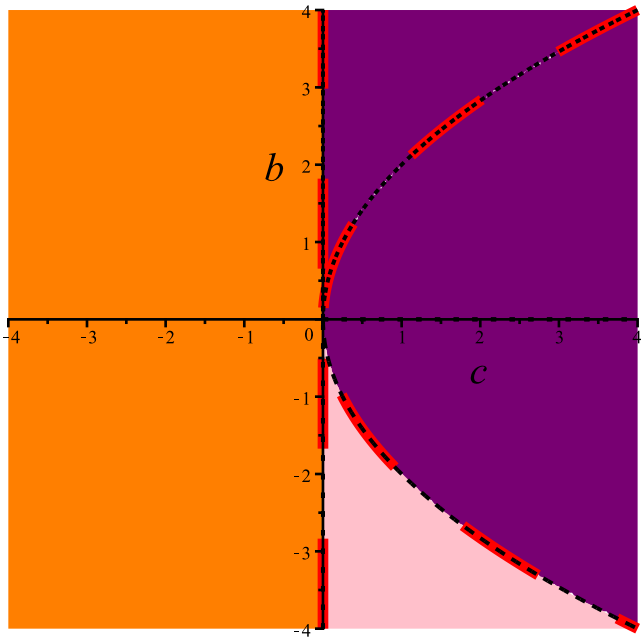
```
> discriminantCurve2 := plots:-implicitplot(discriminantPart2 =
0, c=-4..4, b=-4..4, color=red, thickness=8, labels=[c,b],
view=[-4..4,-4..4], labelfont=["NewTimesRoman",24]):
discriminantCurves1and2 := plots:-display(discriminantCurve1,
discriminantCurve2);
```




Part b.

The open CAD is the same as Figure 1 part b, therefore we only add a plot for the final result after solving the system in sample points and counting the number of positive real solutions.

```
> # positive solutions regions
  pregion7 := plots:-inequal(c < 0, c=-4..4, b=-4..4, color=
  coral):
  pregion8 := plots:-inequal({c > 0, b < 0, b^2-4*c > 0}, c=-4.
  .4, b=-4..4, color=pink):
  pregion9 := plots:-inequal(b^2-4*c < 0, c=-4..4, b=-4..4,
  color=purple):
  pregion10 := plots:-inequal({c > 0, b > 0, b^2-4*c > 0}, c=-4.
  .4, b=-4..4, color=purple):
  # fianl
  finalPartition2 := plots:-display(pcurve1, pcurve2, pregion7,
  pregion8, pregion9, pregion10);
```




Computing the resultants using Maple's resultant command.

```
> resultat1 := resultant(f, diff(f,x), x);
```

$$resultat1 := -b^2 + 4\,c \tag{1.6}$$

```
> resultant2 := resultant(f, x, x);
```

$$resultant2 := c \tag{1.7}$$

Computing the Dixon resultant using the Maple package DR.

```
> read("C:\\Home\\Packages\\dixon-master\\dr.mpl"): # replace the
  directory with the one that dr.mpl file is located there on
  your own computer.
> DixonPolynomial1 := DR:-DixonPolynomial(system1, [x])[1];
```

$$DixonPolynomial1 := b^2 + b\,x + x\_\,b + 2\,x\,x\_ - 2\,c \tag{1.8}$$

Note that 'x_' is the auxiliary variable \bar{x}.

```
> DixonMatrix1 := DR:-DixonMatrix(DR:-DixonPolynomial(system1,
  [x]))[1];
```

$$DixonMatrix1 := \begin{bmatrix} b^2 - 2\,c & b \\ b & 2 \end{bmatrix} \tag{1.9}$$

```
> DixonResultant1 := DR:-DixonResultant(system1, [x]);
```

$$DixonResultant1 := -b^2 + 4\,c \tag{1.10}$$

For the second discriminant curve;

```
> DixonPolynomial2 := DR:-DixonPolynomial(system2, [x])[1];
  DixonMatrix2 := DR:-DixonMatrix(DR:-DixonPolynomial(system2,
  [x]))[1];
  DixonResultant2 := DR:-DixonResultant(system2, [x]);
```

$$DixonPolynomial2 := x\,x\_ - c$$

$$DixonMatrix2 := \begin{bmatrix} -c & 0 \\ 0 & 1 \end{bmatrix}$$

$$DixonResultant2 := -c \tag{1.11}$$

# Appendix: resultant may include extra components.

An example of a case where resultant provides extra components even in case of 2 polynomials in 1 variable. This is stated in the third paragraph of section 5.1 of the paper.

```
> f__1 := a * x + 1;
  f__2 := a * x + b;
```

$$f_1 := a\,x + 1$$

$$f_2 := a\,x + b \tag{2.1}$$

```
> g := resultant(f__1, f__2, x);
```

$$g := a\,b - a \tag{2.2}$$

```
> factor(g);
```

$$a\,(b - 1) \tag{2.3}$$

So resultant gives us two components a = 0 and b - 1 = 0.

```
> PolynomialIdeals:-EliminationIdeal(PolynomialIdeals:-
  PolynomialIdeal([f__1, f__2]), {a,b});
```

$$\langle b - 1 \rangle \tag{2.4}$$

But the discriminant variety is just b - 1 = 0. So a = 0 was an extra component in the output of the resultant.

End of the file.