

This is far from a documentation for PtCut

Christoph Lüders

January 21, 2020

Abstract

PtCut is a Python program to calculate tropical prevarieties and tropical equilibria. This is its documentation. Well, it's a sad excuse for one.

1 Command line options

- **-eX, ..., --ep X, ...**
Specify the reciprocal of the ε parameter. Default is **-e5**, i.e., $\varepsilon = 1/5$. You can specify integers or fixed point float values (like **-e1.1**). The value **--ep pX** signifies the smallest prime number larger than 10^X , i.e., **p1** is 11, **p2** is 101, **p3** is 1009, etc. The value **--ep cX** signifies the smallest prime number larger than 10^X plus 1, a composite number. The value **--ep eX** signifies 10^X .
You can specify multiple values for ε , separated by commas (but no spaces). Calculations will then be performed for each values successively.
Note: all those values specify the reciprocal of ε !
Note: there is only one '-' and no space for -e!
Note: there must be no spaces between different values for ε !
- **--seed X**
Specify the random seed that is e.g. used for **--filter** and **--shuffle**. If not specified, a seed by the operating system is used. Specifying the same seed will lead to the same random choices. The seed used is printed prior to calculation.
- **--st**
Switch on timing of intersections and inclusion tests. Can cost some 20% run-time.
- **-s**
Switch on extensive logging of intersections and inclusion tests. This can slow calculation down by a large factor.
- **-p**
Print the solution in **.lp** format to the screen.
- **--stl**
Print the solution in **.lp** format to the log file.
- **--no X, Y, ...**
Remove model(s) X , Y , etc. from list of models to calculate. Enables you to use **--all --no BIOMD0000000026** to calculate all models, but Biomodel 28.
Note: there must be no spaces between model names!
- **--sumup, --sum**
When handling equations, substituting the parameters will re-evaluate the formulas and collect terms with matching variables.
Example: assume $k_1 = 2$, $k_2 = 3$. Then $k_1x_1 + k_2x_1$ becomes $2x_1 + 3x_1$ (intermediate) and $5x_1$ in the end. Even more dramatic, $k_1x_1 - k_2x_1$ will collapse to 0 if $k_1 = k_2$.
Default is to switch off the **sumup** feature, thus creating multiple points after tropicalization. **--sumup** requires **--keep-coeff**.

- **--keep-coeff, --keep-coeffs**
When handling the equations, literal parameters will be not be ignored (parameters listed in `Params.txt` are never ignored).
Example: in the term $2k_1x_1$, the “2” will not be ignored (the value of k_1 is never ignored).
Default is to ignore literal parameters.
- **--merge-param, --merge-params**
When handling the equations, just substitute the parameters and then do the logarithm and rounding.
By default, the logarithm and rounding are calculated per parameter separately, thus rounding errors might add up.
- **-C**
Switch into “Chris mode”, that is a shorthand for `--sumup, --keep-coeff, --merge-params`.
- **--bb, --bbox**
Switch on bounding boxes: in common planes checking, PtCut will calculate bounding boxes for all polyhedra. These individual bounding boxes are then joined to form the bounding box of their bag. The bag bounding boxes of all bags are then intersected. This total bounding box is the space in which all solutions must lie and thus, all polyhedra are intersected with it.
This can dramatically reduce the number of possible combinations. However, the calculation needs V-representation and it takes time to compute that. This means that it’s not always efficient to calculate bounding boxes.
- **--filter X**
After each iteration, limit the number of temporary polyhedra to X . If there are more than X polyhedra, a random selection of them is discarded (actually, they are not calculated in the first place). Obviously, this saves a lot of time, but computes only a subset of the solution.
- **--remove X,...**
Remove named polyhedra from bags prior to calculation. The respective names have the form $N.M$, where N is the number of the bag (starting at 0) and M is the number of the polyhedron in the bag (starting at 0). Those are the same numbers that are listed when common planes calculation finds superfluous polyhedra and lists them as **Removed:** (see log file or use `-v` option to see this output on the screen).
If only polyhedra are removed that were listed by common planes calculation, this will *not* change the solution. So, one can run PtCut multiple times, with and without certain parameters (like `--box, --nc, --one, --common2`, different `-a` or `-l` values) and collect superfluous polyhedra and use them with `--remove` to reduce the number of combinations.
Note: there must be no spaces between polyhedra names!
- **--nc**
Switch off common places calculation. Sometimes common planes calculation causes polyhedra to be too complex and calculations can get extremely slow.
- **--one**
Only bags with one polyhedron are intersected into all other bags while calculating common planes.
- **--common2**
Only bags with one polyhedron and common hyperplanes (but no common half-spaces) are intersected into all other bags while calculating common planes.
- **--shuffle**
Randomly shuffle the order of bags at the beginning. If a random order of intersections is desired, you must add `-10` option as well to switch off *likeness* calculations when selecting the next bag for intersection.

- **--runs X**
Perform X runs of the same model(s). This makes sense when used with **--shuffle** and **--maxruntime**, since each time a different order will be used.
- **--maxruntime X**
Stop computation after X seconds. Useful with **--shuffle** and **--runs**.
- **--all**
--simple
--easy
--fast
--slow
--hard
--bball
Select a list of models to calculate. There are certain relations:

$\text{simple} = \text{fast} + \text{slow}$
 $\text{all} = \text{simple} + \text{hard}$
 $\text{all} \supset \text{simple} \supset \text{easy} \supset \text{fast}$
 $\text{all} \supset \text{bball}$

The list **bball** leaves out models that are slow when run with **--bbox** option.y

- **--nonewton**
Build no Newton polytope to compute and filter out interior points. Instead, just use all points, since superfluous points will be removed anyway in the process.
- **--cc, --concomp**
Calculate the number of connected components of the polyhedra in the prevariety. See **--contype X** for what exactly “connected” means.
- **--contype X**
When are two polyhedra considered “connected”? Several types are supported:
 - Type 0: The intersection of both polyhedra is non-empty.
 - Type 1: The intersection of both polyhedra is non-empty and its dimension is lower than both polyhedra, or a point.
 - Type 2: Both polyhedra have the same dimension and their intersection is non-empty.
- **--multiple-lpfiles**
Instead of saving the solution into one **.lp** file, separated by lines starting with 40 backslashes (multi-lp files), save each polyhedron in its own **.lp** file.
- **--nosave**
Don't save the solution to one or multiple **.lp** files.
- **--pv X, \dots**
--paramvars X, \dots
--param_vars X, \dots
Treat parameters X, \dots as variables, even though they might be defined in **Params.txt**.
- **--lbound X, \dots**
--logbound $X : l : u, \dots$
Bound each variable X as $10^l \leq X \leq 10^u$. You can leave off one (or both) bounds by not specifying them, like **--lbound $x1:2$** or **--lbound $x1::2$** . The bounds are done by adding one additional bag with one polyhedron that has half-spaces for the specified variables.
Note: There are no spaces between the commas or colons!
- **--stdlbound X, \dots**
--stdlogbound $X:u, \dots$
Bound each parameter-as-variable X as $x \cdot 10^{-u} \leq X \leq x \cdot 10^u$, where x is the standard value from **Params.txt**. You can use **all** instead of X to bound all

parameters-as-variable. The bounds are done by adding one additional bag with one polyhedron that has half-spaces for the specified variables.

Note: There are no spaces between the commas or colons!

- `--maxmem X`
Limit the amount of physical memory the process can use to X bytes. Units K, M, G, T and P are supported.
- `--grid`
Do a grid search over changing parameters. You can specify more than one parameter and the range and step size in which they should change and if the step size is additive or multiplicative.

Examples:

- `ptcut BIOMD0000000026 --grid k17:80:200:10 --stl`
This will set `k17` to values between 80 and 200 in steps of 10.
- `ptcut BIOMD0000000026 --grid k1:1e-4:1e2:*10 --stl`
This will set `k1` to values between 10^{-4} and 10^2 , where the value is multiplied by 10 in each step.
- `ptcut BIOMD0000000026 --grid k1:1e-4:1e2:*10,k2:1e-4:1e4:*10 --stl`
This will change the values of `k1` and `k2` over a grid of values. Remember to enclose the parameter to `--grid` in quotes on Linux/Mac if it contains an asterisk.

To save the results, i.e. the polyhedra, of each computation, you have to specify the `--stl` option, which will put them in the log file.

2 Running PtCut

To get proper output about the machine and CPU used, please install `py-cpuinfo`. With Anaconda under Windows, this can be done by entering this command:

```
conda install -c brian-team py-cpuinfo
```