Using HyperTraPS: current status

Pablo Herrera Nieto

Using HyperTraPS from CPM-SSWM-Sampling repo

TLDR

- **TODO**: Modify hypegraph plot to include labels of all genotypes.
- The output gives all the accesible genotypes: so we do not need to compute them.
- There are many parameters for running everything: the easiest is to stick to defaults and check the convergence.
- Besides default output, I am saving the transition

Example

A simple example:

```
## ((A AND B) or C) to reach D

dB_c2 <- matrix(
  c(
      rep(c(1, 0, 0, 0), 300) #A
    , rep(c(0, 1, 0, 0), 300) #B
    , rep(c(0, 0, 1, 0), 300) #C
    , rep(c(1, 1, 0, 0), 200) #AB
    , rep(c(1, 0, 1, 0), 200) #AC
    , rep(c(0, 1, 1, 0), 200) #BC
    , rep(c(1, 1, 1, 0), 100) #ABC
    , rep(c(1, 1, 0, 1), 200) #ABD
    , rep(c(0, 0, 1, 1), 250) #CD
    , rep(c(1, 1, 1, 1), 200) # ABCD
    , rep(c(0, 0, 0, 0), 10) # WT
  ), ncol = 4, byrow = TRUE
)
colnames(dB_c2) <- LETTERS[1:4]

do_HyperTraPS(dB_c2, "HP_c2", runs = 500, bi=200)
```

It takes as parameters:

1. the data (it can also be the name of csv file),
2. the name of the folder to generate it (if NULL it will run in `/tmp`)
3. some paremeters to control the run: more could (and should) be included
4. `dry_run = TRUE` only executes the analysis section of the pipeline.

Let's see some output!

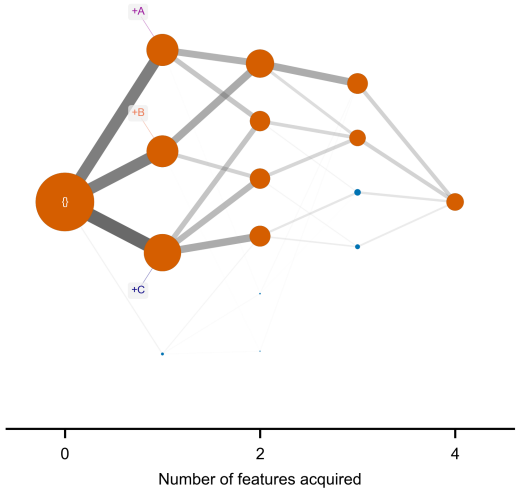

Figure 1: hypergraph

Ok. Here we see how some mutations are built on top of previous ones. But ... I do not understand it. For example, when we have 2 mutations, what is the order of the genes? It is not ordered as A-B-C-D, because A–>A is not no possible? So, what is the order?

This plot is generated with the output of the walkers (implementation here) that looks like this.

from	to	weight
0	1	39736
0	2	185
0	4	177
0	8	2
1	3	29618
1	5	110
1	9	8
2	3	182
2	6	3
3	7	19791
3	11	9
4	5	70
4	6	7
5	7	180
6	7	10
7	15	9981
8	9	2
9	11	10
11	15	19

- **From** and **To**: genotype number. This is the number of the vertex of the hypergraph. For example, the transition 7–>15 means: ABC –> ABCD (vertex in an hypergraph are codified in binary, so, 7 –> 0111 –> ABC, last position corresponds to A and first to the Lth gene); and 15 –> 1111 –> ABCD); another, the transition 1 –> 3 means: A –> AB (1 –> 0001 –> A and 3 –> 0011 –> AB).

- **Weigths**: counts in the transition.

So, the picture above represents jumps between genotypes with one more mutation every time, but the path is not clear. I am fixing it.

One last thing, given the above output we can directly compute transition probabilities between genotypes, so we do not need to compute them (we save ourselves from deciding if this is an AND or OR model).

Installation

You need to have a running conda environment and **activate** it in the terminal with `conda activate <env_name>`.

To install just follow the instructions of the `README.md` in the HyperTraPS repo:

> ./configure.sh conda env create –name HyperTraPS –file conda_requirements.yml
>
> From: https://github.com/sgreenbury/HyperTraPS/blob/master/README.md

Then, `bin` and `src/python` from the HyperTraPS folfer must be included in the `PATH`. Finally, `bin` from the `CPM-SSWM-Sampling` folder also must be added to `PATH`.

I have done it in OncoStation and in Draco, but only for my user. I do not know if doing a system-wide installation is reasonable: HyperTraPS runs on `python2` (only God knows why!) and with the conda env is cleaner (imho).

Implementation details

- **Setting**: creates directory and stores data there.
- **Execution of HyperTraPS**: from `hypertraps_process.R::do_HyperTraPS`.

```
print("Converting Data")
system("convert_data_to_transitions.py -data data.csv -input_type cross-sectional")

### MCM sampler
## TODO check the meaning of all parameters and allow to set them
print("Sampling posterior")
time_posterior <- system.time(
  system(sprintf("RUN_MCMC_SAMPLER -f transitions.txt
  -M second-order -N %1.f -b %1.f -r %1.f -S %1.f",
   runs, bi, r, seed))
)["elapsed"]
cat("Elapsed time for sampling posterior ", time_posterior, "\n")

### Run walkers
print("Generating Paths")
system(sprintf("RUN_PW -w match-data -b %1.f -R 100", bi))
system(sprintf("RUN_PW -w zero-one -b %1.f -R 100", bi))
```

System calls of the executables of HyperTraPS. Those are implemented in C++.

- **Analysis and Plotting**:

```
system(sprintf("plot_mc_stats.py -b %1.f", bi))
system(sprintf("plot_hypercube_graph.py -f 'forwards_list-pord-match-data.csv'
  -outfile_graph 'forwards-hypercube-graph-mach-data-g0'
```

```
  -transition_data 'transitions.txt'
  -labels 'labels.csv' -label_type 'greedy_data'
  -labels_fontsize 4 -layout_type 'spring' -aspect 0.9
  -width 3.4 -out_type 'png'"))

system(sprintf("plot_feature_graph.py -f 'forwards.txt' -layout_type 'circular'
-prob_type 'conditional' -data_type 'match-data' -width 4
-fontsize 10 -any_time 0 -node_size 100
-connection_style 'arc3,rad=-0.3' -outfile_type 'png'"))

system(sprintf("plot_ordering_histogram.py -f 'forwards_list-pord-zero-one.csv'
  -f2 'forwards_list-pord-match-data.csv'
  -transition_data 'transitions.txt' -labels 'labels.csv'
  -fontsize 6 -xevery 1 -aspect 0.9
  -verbose 'no' -outfile 'forwards-ws1-ws2' -out_type 'png'"))
```

Changes respect the examples they include in the `HyperTraPS/notebooks/HyperTraPS_tutorial.ipynb`:

`+ \`plot_feature_graph\`: \`-prob_type conditional\` instead of \`-prob_type joint\``

- **Custom output**: the systems calls above generate a bunch of plots. In the last step of the pipeline I reuse some of their code to extract some data:
  - Transition matrix from gene to gene: this matrix can be computed as as joint or conditional probability. The change above to conditional probability was done to match the output from others CPMs in the `CPM-SSWM-Sampling` repository.
  - Transtions between accesible genotypes: see next section.

Here we can include what we want and is WIP.

Most relevant outputs

Open Questions