Probabilistic Gene Nets

Project 6

General Structure

pbndriver

[pnet,rnet] = ruletree(rule,prob)

New

STM = genestm(wire,rule)

d2b

b2d

pbndriver

1. Set wire, rule, prob
2. Call ruletree to output pnet and rnet

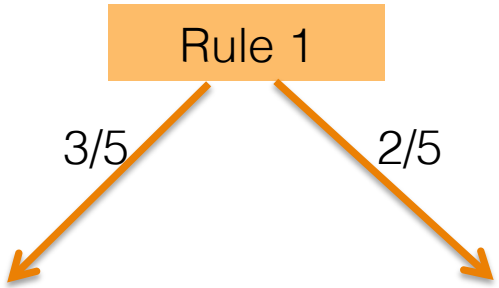
… [we now pause to examine the mechanics of ruletree]

3. Build STM from pnet and rnet and view it using biograph

[pnet, rnet]=ruletree(rule,prob)

This is still “rule 1” (one rule), it can just take two forms, with different associated probabilities.

gene	wire	rule	probability
1	1 2 3	238, 230	3/5, 2/5
2	1 2 3	182	1
3	1 2 3	232, 128	1/2, 1/2



Within the function ruletree, we will be working primarily with a large adjacency matrix, ptree. It it neither an input nor an output, but is used within the function to find pnet and rnet, and is imaged using biograph to provide the Rule Tree figure.

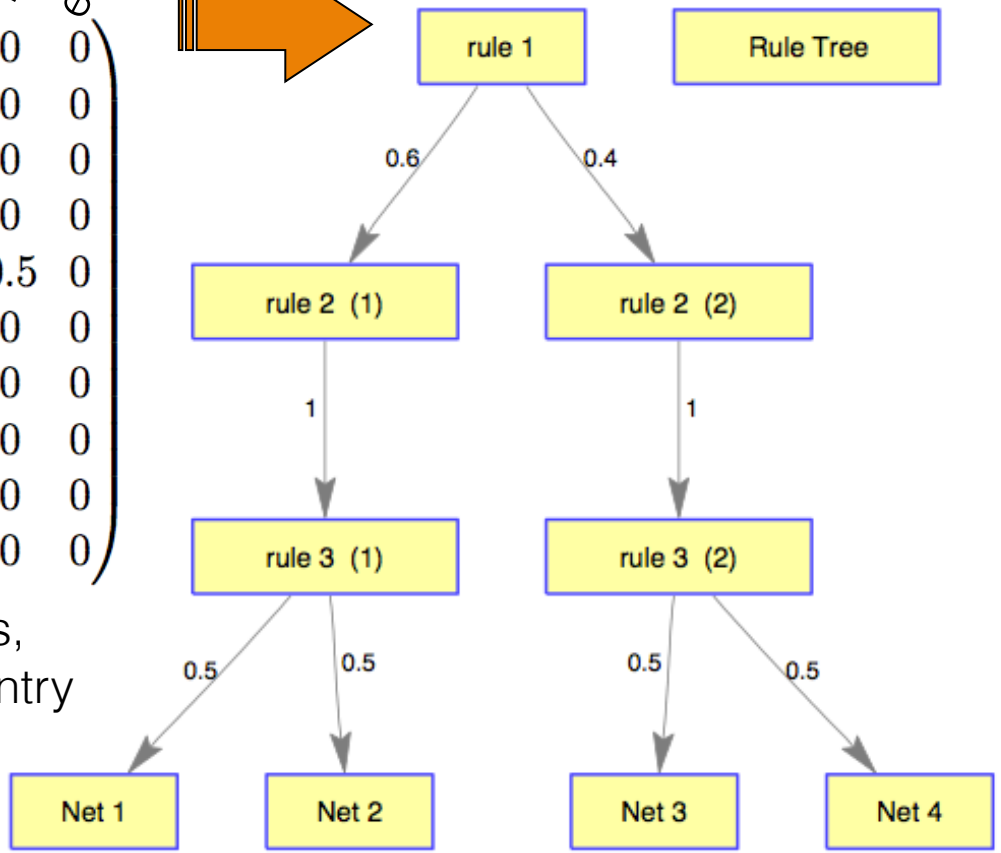
[ruletree function]

Building the Rule Tree

ptree

	node 1	node 2	node 3	node 4	node 5	node 6	node 7	node 8	node 9	extra col
node 1	0	0.6	0.4	0	0	0	0	0	0	0
node 2	0	0	0	1	0	0	0	0	0	0
node 3	0	0	0	0	1	0	0	0	0	0
node 4	0	0	0	0	0	0.5	0.5	0	0	0
node 5	0	0	0	0	0	0	0	0.5	0.5	0
node 6	0	0	0	0	0	0	0	0	0	0
node 7	0	0	0	0	0	0	0	0	0	0
node 8	0	0	0	0	0	0	0	0	0	0
node 9	0	0	0	0	0	0	0	0	0	0
extra row	0	0	0	0	0	0	0	0	0	0

biograph



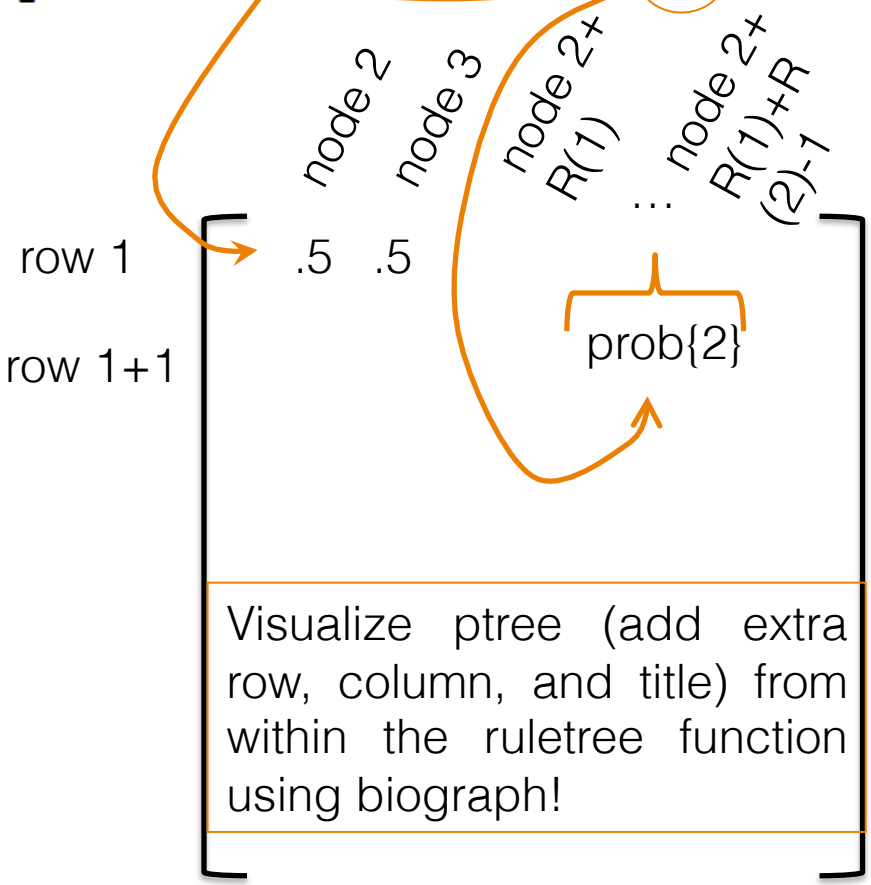
In previous adjacency matrices, we've placed a "1" in the (i,j) entry if nodes i and j are connected. Now, we place a probability.

How can we automate this process using the information in rule and prob?

Building the adjacency matrix, **ptree**

Input:

```
rule = {[231 230], 64, [5 7], 108, 61, [62 60]};
prob = {[1/2 1/2], 1, [1/2 1/2], 1, 1, [1/2 1/2]};
```



Visualize ptree (add extra row, column, and title) from within the ruletree function using biograph!

```
ptree(1,2:1+R(1)) = prob{1}.
```

Handling the ids using num2str

If "i" is a counter you want to convert to a string:

ids{ind} = ['rule ' num2str(i)];

You will also have to account for different expressions of each rule, e.g. rule 2 (1)

pnet

- **pnet** contains the probabilities of arriving at each of the final nodes in the ruletree.
- You can implement this step-wise in MATLAB: accumulate a prob cell that contains the probabilities of arriving at the nodes in each "layer"
- For example stepprob={[each node in layer 1]
  [each node in layer 2]…
  [each node in final layer]}

In this case, what is **pnet**?

rnet

- **rnet** is “a vector of N rule indicators”
- → rnet tells us the sequence of rules used to arrive at each of our final nodes in the rule tree.

```
rnet <- 111111 111112 112111 112112 211111 211112 212111 212112
```

gene net 1 ..

Here you are just tracing paths down ruletree.
1=used the first expression of a rule (left path)
2=used the second expression of a rule (right path)

- Let’s build by hand for our small example.
- How to code: use powers of 10?
- Could same way as pnet: build cell then take final entry which corresponds with terminal nodes.

a better **rnet**

- **rnet** is "a *matrix* of N rule indicators"
- → rnet tells us the sequence of rules used to arrive at each of our final nodes in the rule tree.

rnet = [238, 182, 232; } rule sequence for gene net 1 …
238, 182, 128;
230, 182, 232;
230, 182, 128];

pbndriver

1. Set wire, rule, prob
2. Call ruletree to output pnet and rnet

3. Build STM from pnet and rnet and view it using biograph

Building STM

- Run last week's genestm on **wire** and the different rule combinations.

  → now it's easy to get a rule set for a particular gene net: just use that row of **rule!**
- genestm will output an STM. What is the probability that this STM will occur? Where do we find it?
- The final STM is simply the sum of all possible STMs generated by genestm, weighted by the probabilities that they will occur, found in the corresponding elements of **pnet**.