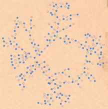
Applications of Random Networks

Complex Networks
CSYS/MATH 303, Spring, 2011

Prof. Peter Dodds

Department of Mathematics & Statistics
Center for Complex Systems
Vermont Advanced Computing Center
University of Vermont

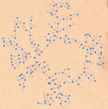


Outline

Analysis of real networks

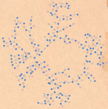
- How to build revisited
- Motifs

References



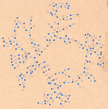
More on building random networks

- Problem: How much of a real network's structure is non-random?
- Key elephant in the room: the degree distribution P_k.
- First observe departure of P_k from a Poisson distribution.
- Next: measure the departure of a real network with a degree frequency N_k from a random network with the same degree frequency.
- Degree frequency N_k = observed frequency of degrees for a real network.
- What we now need to do: Create an ensemble of random networks with degree frequency N_k and then compare.



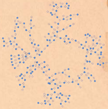
More on building random networks

- Problem: How much of a real network's structure is non-random?
- Key elephant in the room: the degree distribution P_k.
- First observe departure of P_k from a Poisson distribution.
- Next: measure the departure of a real network with a degree frequency N_k from a random network with the same degree frequency.
- Degree frequency N_k = observed frequency of degrees for a real network.
- What we now need to do: Create an ensemble of random networks with degree frequency N_k and then compare.



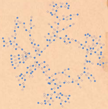
More on building random networks

- Problem: How much of a real network's structure is non-random?
- Key elephant in the room: the degree distribution P_k.
- First observe departure of P_k from a Poisson distribution.
- Next: measure the departure of a real network with a degree frequency N_k from a random network with the same degree frequency.
- Degree frequency N_k = observed frequency of degrees for a real network.
- What we now need to do: Create an ensemble of random networks with degree frequency N_k and then compare.



More on building random networks

- Problem: How much of a real network's structure is non-random?
- Key elephant in the room: the degree distribution P_k.
- First observe departure of P_k from a Poisson distribution.
- Next: measure the departure of a real network with a degree frequency N_k from a random network with the same degree frequency.
- Degree frequency N_k = observed frequency of degrees for a real network.
- What we now need to do: Create an ensemble of random networks with degree frequency N_k and then compare.



More on building random networks

- Problem: How much of a real network's structure is non-random?
- Key elephant in the room: the degree distribution P_k.
- First observe departure of P_k from a Poisson distribution.
- Next: measure the departure of a real network with a degree frequency N_k from a random network with the same degree frequency.
- Degree frequency N_k = observed frequency of degrees for a real network.
- What we now need to do: Create an ensemble of random networks with degree frequency N_k and then compare.

More on building random networks

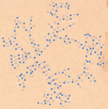
- Problem: How much of a real network’s structure is non-random?
- Key elephant in the room: the degree distribution P_k.
- First observe departure of P_k from a Poisson distribution.
- Next: measure the departure of a real network with a degree frequency N_k from a random network with the same degree frequency.
- Degree frequency N_k = observed frequency of degrees for a real network.
- What we now need to do: Create an ensemble of random networks with degree frequency N_k and then compare.

Outline

Analysis of real networks

- How to build revisited
- Motifs

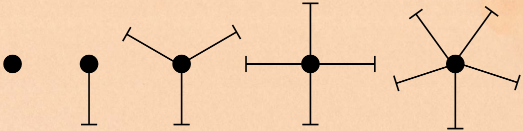
References

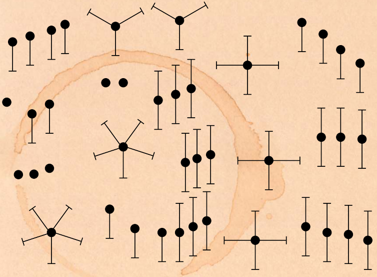


Building random networks: Stubs

Phase 1:

- Idea: start with a soup of unconnected nodes with stubs (half-edges):



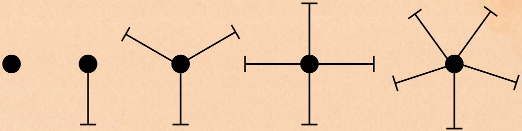


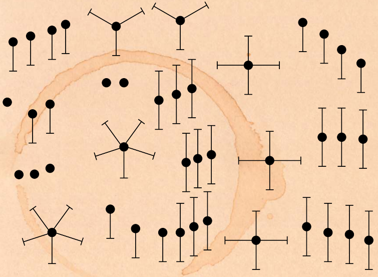
- Randomly select stubs (not nodes!) and connect them.
- Must have an even number of stubs.
- Initially allow self- and repeat connections.

Building random networks: Stubs

Phase 1:

- Idea: start with a soup of unconnected nodes with stubs (half-edges):



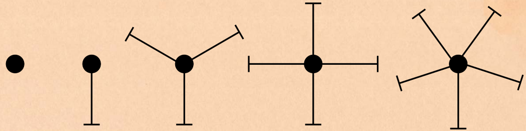


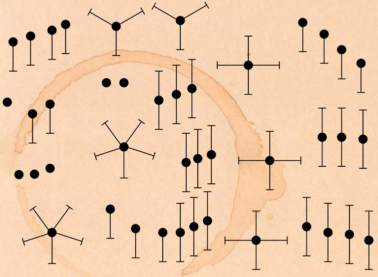
- Randomly select stubs (not nodes!) and connect them.
- Must have an even number of stubs.
- Initially allow self- and repeat connections.

Building random networks: Stubs

Phase 1:

- Idea: start with a soup of unconnected nodes with stubs (half-edges):



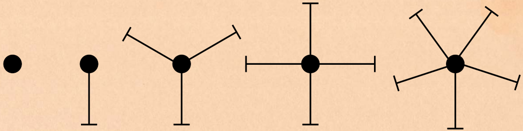


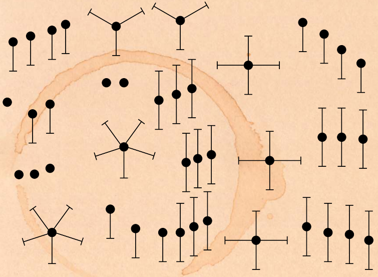
- Randomly select stubs (not nodes!) and connect them.
- Must have an even number of stubs.
- Initially allow self- and repeat connections.

Building random networks: Stubs

Phase 1:

- Idea: start with a soup of unconnected nodes with stubs (half-edges):



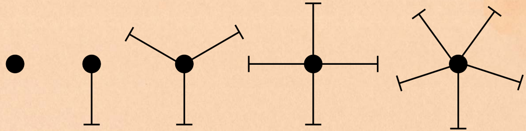


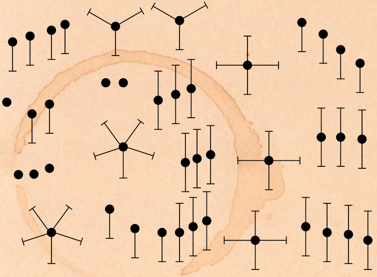
- Randomly select stubs (not nodes!) and connect them.
- Must have an even number of stubs.
- Initially allow self- and repeat connections.

Building random networks: Stubs

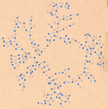
Phase 1:

- Idea: start with a soup of unconnected nodes with stubs (half-edges):





- Randomly select stubs (not nodes!) and connect them.
- Must have an even number of stubs.
- Initially allow self- and repeat connections.

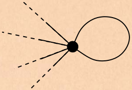


Building random networks: First rewiring

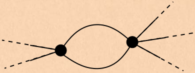
Phase 2:

- Now find any (A) self-loops and (B) repeat edges and randomly rewire them.

(A)



(B)

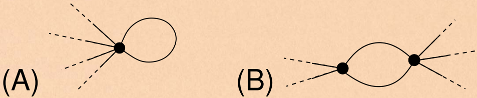


- Being careful: we can't change the degree of any node, so we can't simply move links around.
- Simplest solution: randomly rewire two edges at a time.

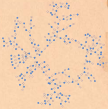
Building random networks: First rewiring

Phase 2:

- Now find any (A) self-loops and (B) repeat edges and randomly rewire them.



- Being careful: we can't change the degree of any node, so we can't simply move links around.
- Simplest solution: randomly rewire two edges at a time.

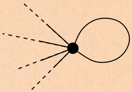


Building random networks: First rewiring

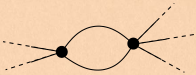
Phase 2:

- Now find any (A) self-loops and (B) repeat edges and randomly rewire them.

(A)

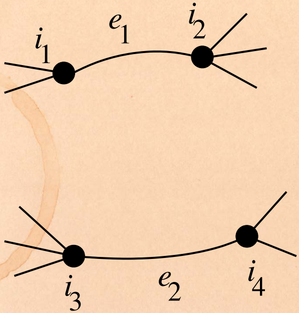


(B)

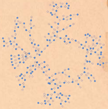


- Being careful: we can't change the degree of any node, so we can't simply move links around.
- Simplest solution: randomly rewire two edges at a time.

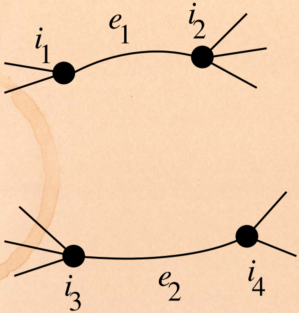
General random rewiring algorithm



- Randomly choose two edges. (Or choose problem edge and a random edge)
- Check to make sure edges are disjoint.
- Rewire one end of each edge.
- Node degrees do not change.
- Works if e_1 is a self-loop or repeated edge.
- Same as finding on/off/on/off 4-cycles. and rotating them.

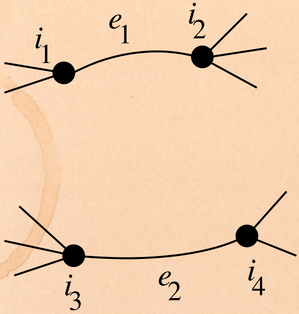


General random rewiring algorithm

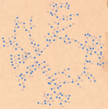


- Randomly choose two edges. (Or choose problem edge and a random edge)
- Check to make sure edges are disjoint.
- Rewire one end of each edge.
- Node degrees do not change.
- Works if e_1 is a self-loop or repeated edge.
- Same as finding on/off/on/off 4-cycles. and rotating them.

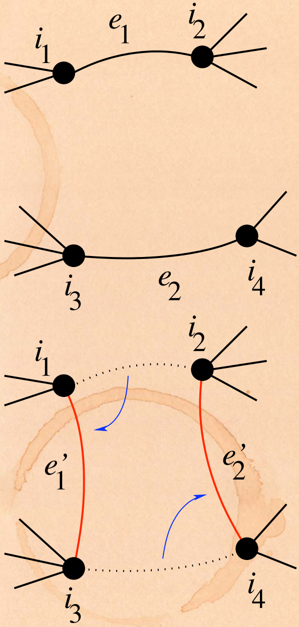
General random rewiring algorithm



- Randomly choose two edges. (Or choose problem edge and a random edge)
- Check to make sure edges are disjoint.
- Rewire one end of each edge.
- Node degrees do not change.
- Works if e_1 is a self-loop or repeated edge.
- Same as finding on/off/on/off 4-cycles, and rotating them.

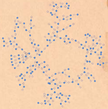


General random rewiring algorithm

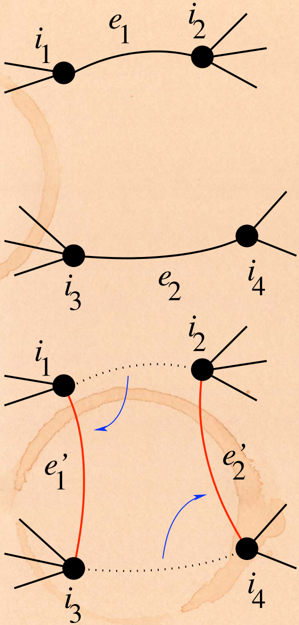


- Randomly choose two edges. (Or choose problem edge and a random edge)
- Check to make sure edges are disjoint.

- Rewire one end of each edge.
- Node degrees do not change.
- Works if e_1 is a self-loop or repeated edge.
- Same as finding on/off/on/off 4-cycles, and rotating them.



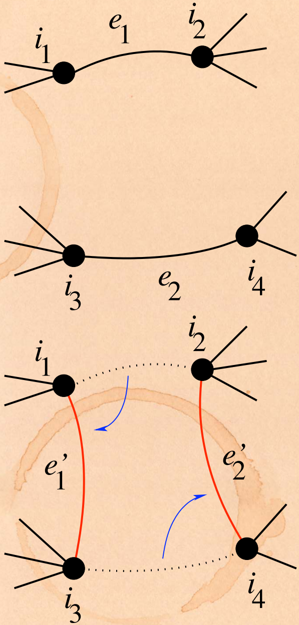
General random rewiring algorithm



- Randomly choose two edges. (Or choose problem edge and a random edge)
- Check to make sure edges are disjoint.

- Rewire one end of each edge.
- Node degrees do not change.
- Works if e_1 is a self-loop or repeated edge.
- Same as finding on/off/on/off 4-cycles, and rotating them.

General random rewiring algorithm



- Randomly choose two edges. (Or choose problem edge and a random edge)
- Check to make sure edges are disjoint.
- Rewire one end of each edge.
- Node degrees do not change.
- Works if e_1 is a self-loop or repeated edge.
- Same as finding on/off/on/off 4-cycles. and rotating them.

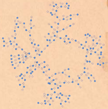
Sampling random networks

Phase 2:

- Use rewiring algorithm to remove all self and repeat loops.

Phase 3:

- Randomize network wiring by applying rewiring algorithm liberally.
- Rule of thumb: # Rewirings $\simeq 10 \times$ # edges [1].



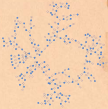
Sampling random networks

Phase 2:

- Use rewiring algorithm to remove all self and repeat loops.

Phase 3:

- Randomize network wiring by applying rewiring algorithm liberally.
- Rule of thumb: # Rewirings $\simeq 10 \times$ # edges[1].



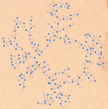
Sampling random networks

Phase 2:

- Use rewiring algorithm to remove all self and repeat loops.

Phase 3:

- Randomize network wiring by applying rewiring algorithm liberally.
- Rule of thumb: # Rewirings $\simeq 10 \times$ # edges [1].

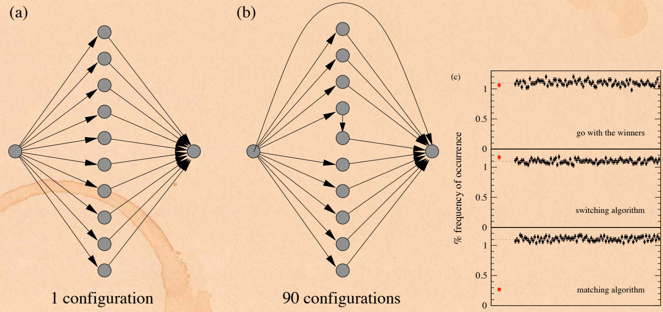


Random sampling

- Problem with only joining up stubs is failure to randomly sample from all possible networks.
- Example from Milo et al. (2003)[1]:

Random sampling

- Problem with only joining up stubs is failure to randomly sample from all possible networks.
- Example from Milo et al. (2003) [1]:

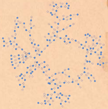


1 configuration

90 configurations

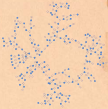
Sampling random networks

- What if we have P_k instead of N_k?
- Must now create nodes before start of the construction algorithm.
- Generate N nodes by sampling from degree distribution P_k.
- Easy to do exactly numerically since k is discrete.
- Note: not all P_k will always give nodes that can be wired together.



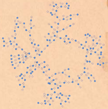
Sampling random networks

- What if we have P_k instead of N_k?
- Must now create nodes before start of the construction algorithm.
- Generate N nodes by sampling from degree distribution P_k.
- Easy to do exactly numerically since k is discrete.
- Note: not all P_k will always give nodes that can be wired together.



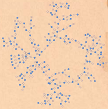
Sampling random networks

- What if we have P_k instead of N_k?
- Must now create nodes before start of the construction algorithm.
- Generate N nodes by sampling from degree distribution P_k.
- Easy to do exactly numerically since k is discrete.
- Note: not all P_k will always give nodes that can be wired together.



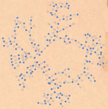
Sampling random networks

- What if we have P_k instead of N_k?
- Must now create nodes before start of the construction algorithm.
- Generate N nodes by sampling from degree distribution P_k.
- Easy to do exactly numerically since k is discrete.
- Note: not all P_k will always give nodes that can be wired together.



Sampling random networks

- What if we have P_k instead of N_k?
- Must now create nodes before start of the construction algorithm.
- Generate N nodes by sampling from degree distribution P_k.
- Easy to do exactly numerically since k is discrete.
- Note: not all P_k will always give nodes that can be wired together.

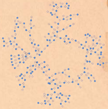


Outline

Analysis of real networks

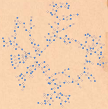
- How to build revisited
- Motifs

References



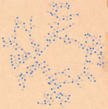
Network motifs

- Idea of motifs [2] introduced by Shen-Orr, Alon et al. in 2002.
- Looked at gene expression within full context of transcriptional regulation networks.
- Specific example of Escherichia coli.
- Directed network with 577 interactions (edges) and 424 operons (nodes).
- Used network randomization to produce ensemble of alternate networks with same degree frequency N_k.
- Looked for certain subnetworks (motifs) that appeared more or less often than expected



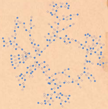
Network motifs

- Idea of motifs [2] introduced by Shen-Orr, Alon et al. in 2002.
- Looked at gene expression within full context of transcriptional regulation networks.
- Specific example of Escherichia coli.
- Directed network with 577 interactions (edges) and 424 operons (nodes).
- Used network randomization to produce ensemble of alternate networks with same degree frequency N_k.
- Looked for certain subnetworks (motifs) that appeared more or less often than expected



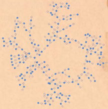
Network motifs

- Idea of motifs [2] introduced by Shen-Orr, Alon et al. in 2002.
- Looked at gene expression within full context of transcriptional regulation networks.
- Specific example of Escherichia coli.
- Directed network with 577 interactions (edges) and 424 operons (nodes).
- Used network randomization to produce ensemble of alternate networks with same degree frequency N_k.
- Looked for certain subnetworks (motifs) that appeared more or less often than expected



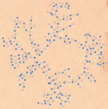
Network motifs

- Idea of motifs [2] introduced by Shen-Orr, Alon et al. in 2002.
- Looked at gene expression within full context of transcriptional regulation networks.
- Specific example of Escherichia coli.
- Directed network with 577 interactions (edges) and 424 operons (nodes).
- Used network randomization to produce ensemble of alternate networks with same degree frequency N_k.
- Looked for certain subnetworks (motifs) that appeared more or less often than expected



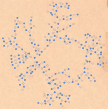
Network motifs

- Idea of motifs [2] introduced by Shen-Orr, Alon et al. in 2002.
- Looked at gene expression within full context of transcriptional regulation networks.
- Specific example of Escherichia coli.
- Directed network with 577 interactions (edges) and 424 operons (nodes).
- Used network randomization to produce ensemble of alternate networks with same degree frequency N_k.
- Looked for certain subnetworks (motifs) that appeared more or less often than expected

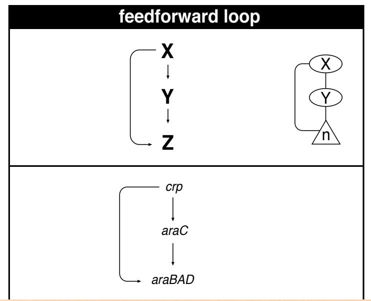


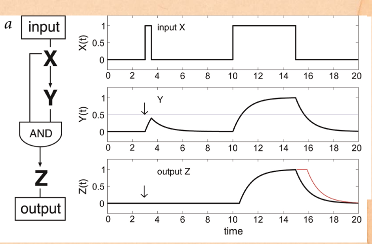
Network motifs

- Idea of motifs [2] introduced by Shen-Orr, Alon et al. in 2002.
- Looked at gene expression within full context of transcriptional regulation networks.
- Specific example of Escherichia coli.
- Directed network with 577 interactions (edges) and 424 operons (nodes).
- Used network randomization to produce ensemble of alternate networks with same degree frequency N_k.
- Looked for certain subnetworks (motifs) that appeared more or less often than expected



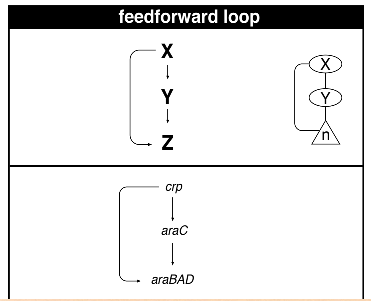
Network motifs

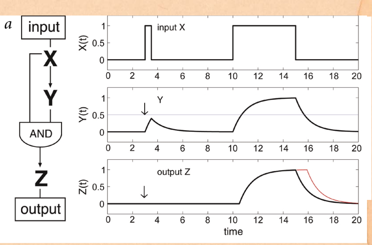




- *Z* only turns on in response to sustained activity in *X*.
- Turning off *X* rapidly turns off *Z*.
- Analogy to elevator doors.

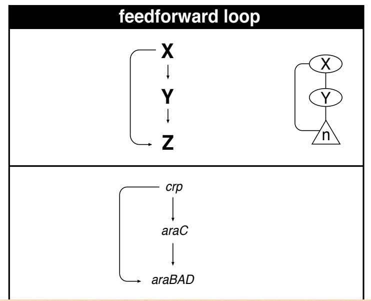
Network motifs

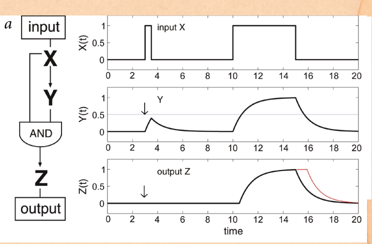




- ► Z only turns on in response to sustained activity in X.
- ► Turning off X rapidly turns off Z.
- ► Analogy to elevator doors.

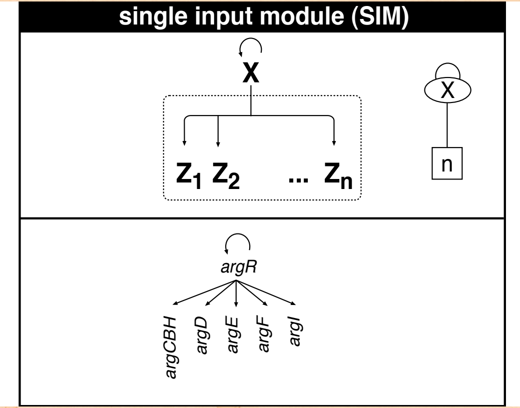
Network motifs





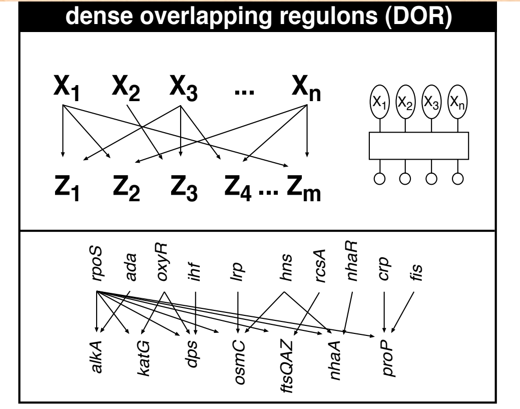
- *Z* only turns on in response to sustained activity in *X*.
- Turning off *X* rapidly turns off *Z*.
- Analogy to elevator doors.

Network motifs



- Master switch.

Network motifs


dense overlapping regulons (DOR)
X_1 X_2 X_3 ... X_n
Z_1 Z_2 Z_3 Z_4 ... Z_m
X_1 X_2 X_3 X_n
rpoS
ada
oxyR
ihf
lrp
hns
rcsA
nhaR
crp
fis
alkA
katG
dps
osmC
ftsQAZ
nhaA
proP

Network motifs

- Note: selection of motifs to test is reasonable but nevertheless ad-hoc.
- For more, see work carried out by Wiggins et al. at Columbia.

Network motifs

- Note: selection of motifs to test is reasonable but nevertheless ad-hoc.
- For more, see work carried out by Wiggins et al. at Columbia.

References I

[1] R. Milo, N. Kashtan, S. Itzkovitz, M. E. J. Newman, and U. Alon.
On the uniform generation of random graphs with prescribed degree sequences, 2003. pdf (⊞)

[2] S. S. Shen-Orr, R. Milo, S. Mangan, and U. Alon.
Network motifs in the transcriptional regulation network of *Escherichia coli*.
Nature Genetics, pages 64–68, 2002. pdf (⊞)