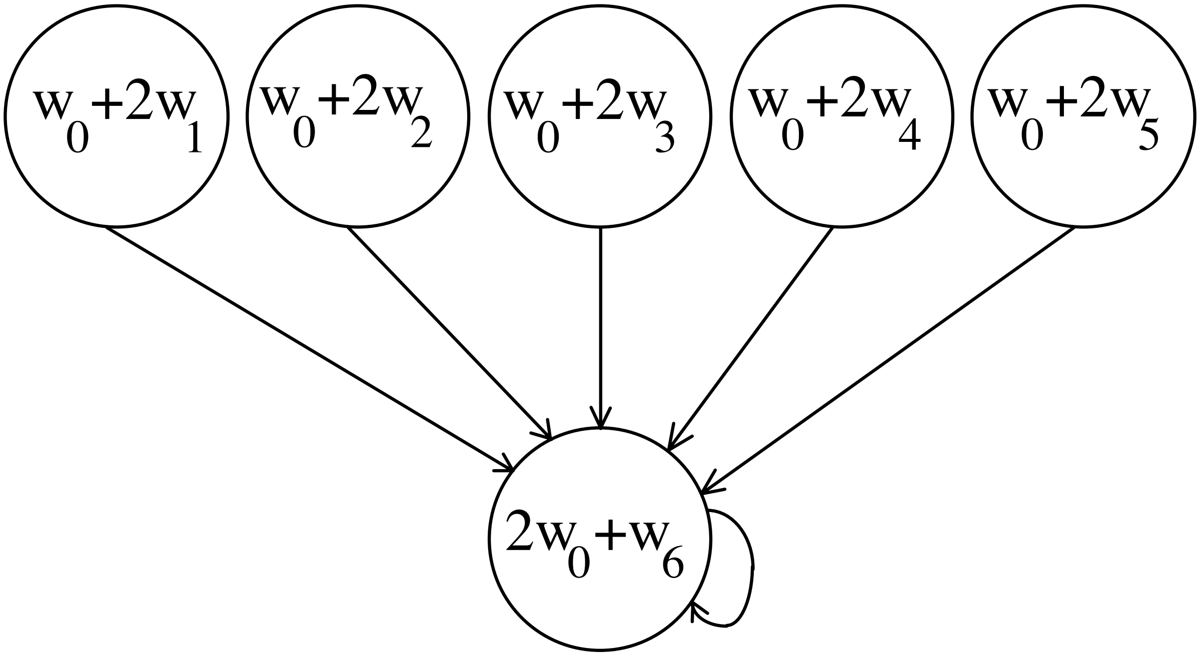
Residual Algorithms:

Reinforcement Learning with Function Approximation

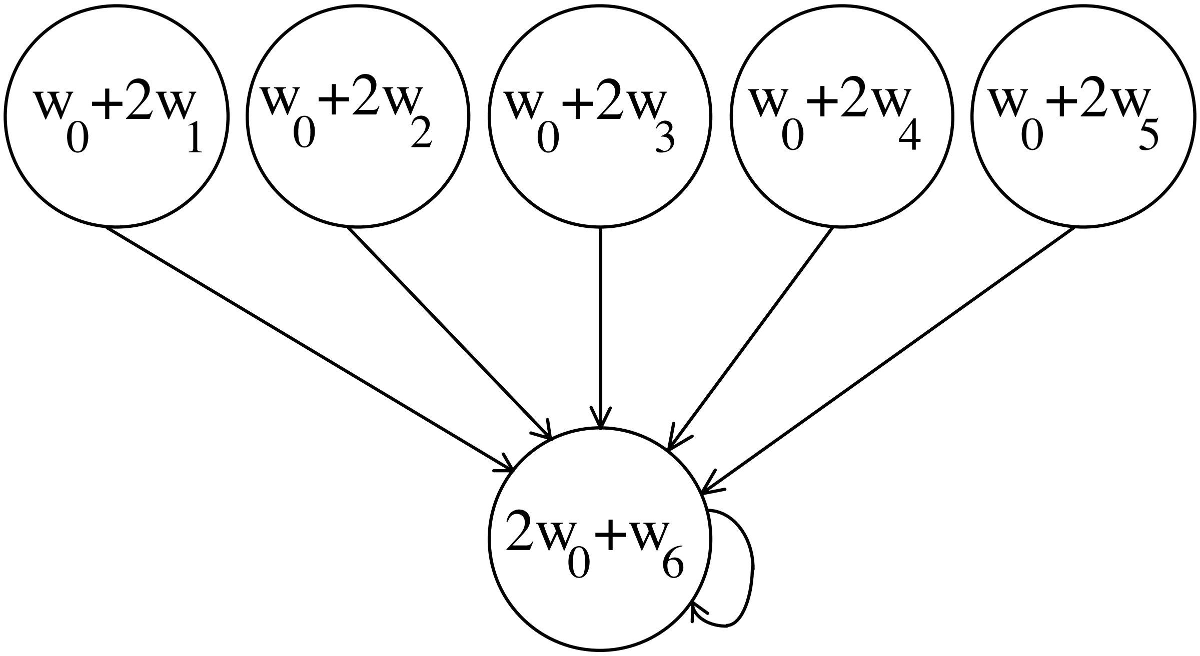
Machine Learning Conference
July 1995

Leemon Baird

U.S. Air Force Academy
baird@cs.usafa.af.mil
http://kirk.usafa.af.mil/~baird



w0+2w1
w0+2w2
w0+2w3
w0+2w4
w0+2w5
2w0+w6



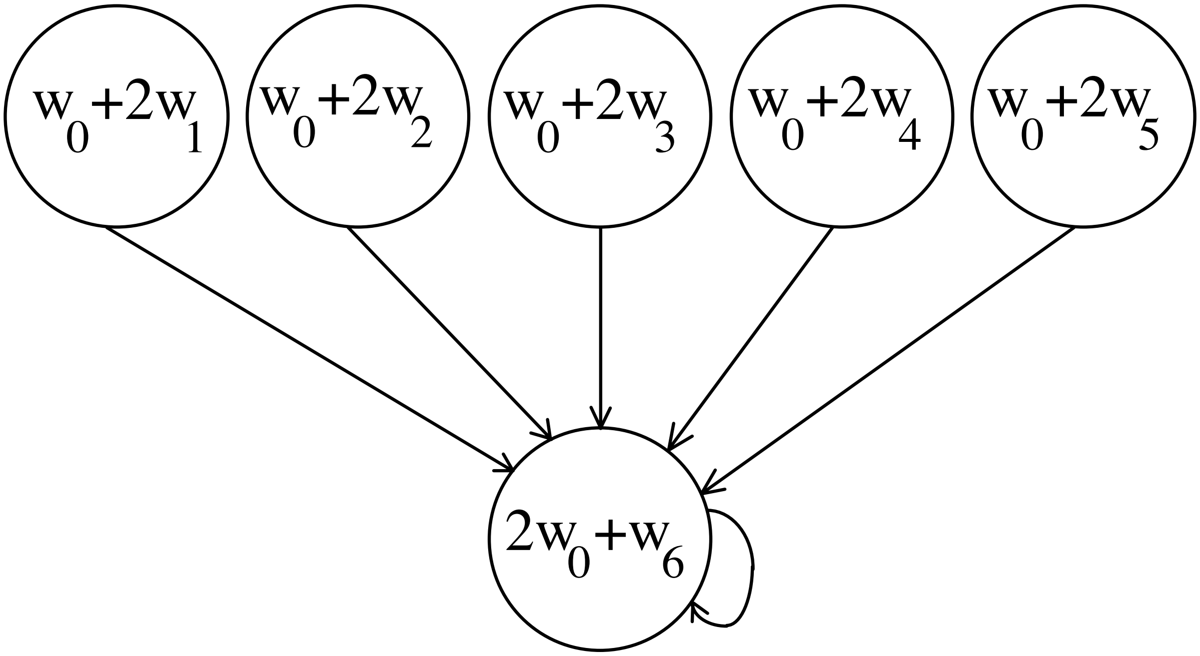
A well-behaved function approximation system:

- All value functions can be represented

- Changing the value of one state with backprop:

  changes neighbors by at most 2/3 as much

- Basically a lookup table plus one generalizing weight (w_0)



Reinforcement learning can fail to converge:

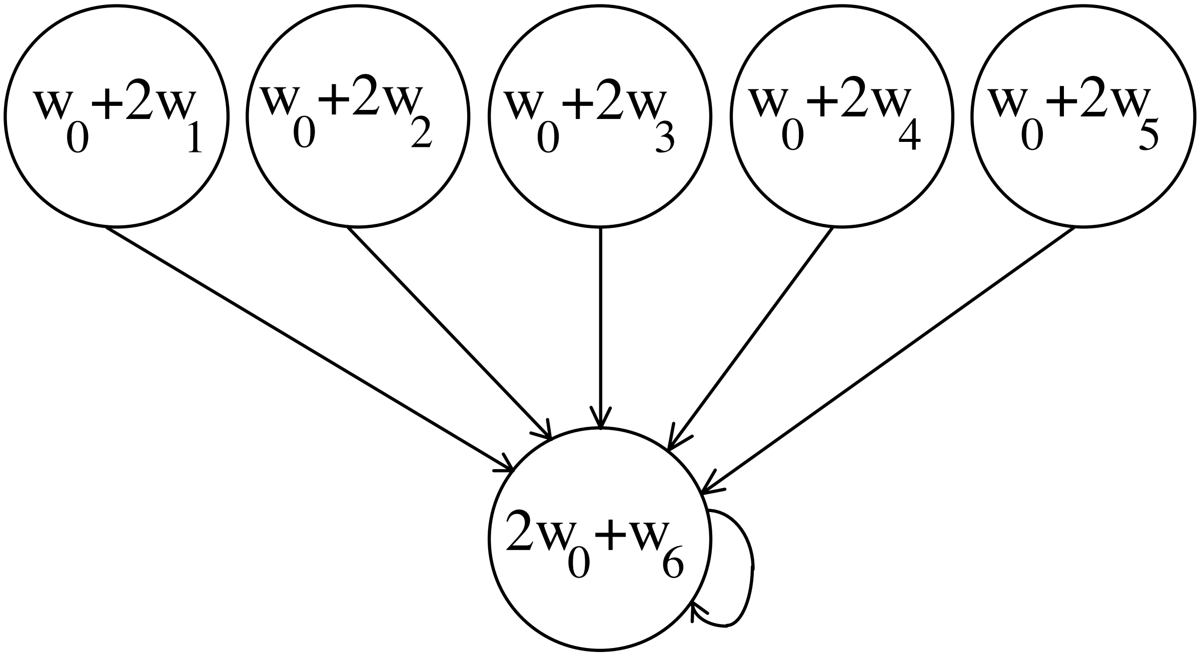
- Learning equation: $\Delta w = -\alpha\left(R + \gamma v_{new} - v_{old}\right)\frac{\partial v_{old}}{w}$

- Every transition updated equally often

- Learning is a special case of TD(0), Q-learning + backprop, and incremental value iteration + backprop

-If state 6 starts high, it climbs more often than falls.

- All states/weights diverge to $\pm\infty$



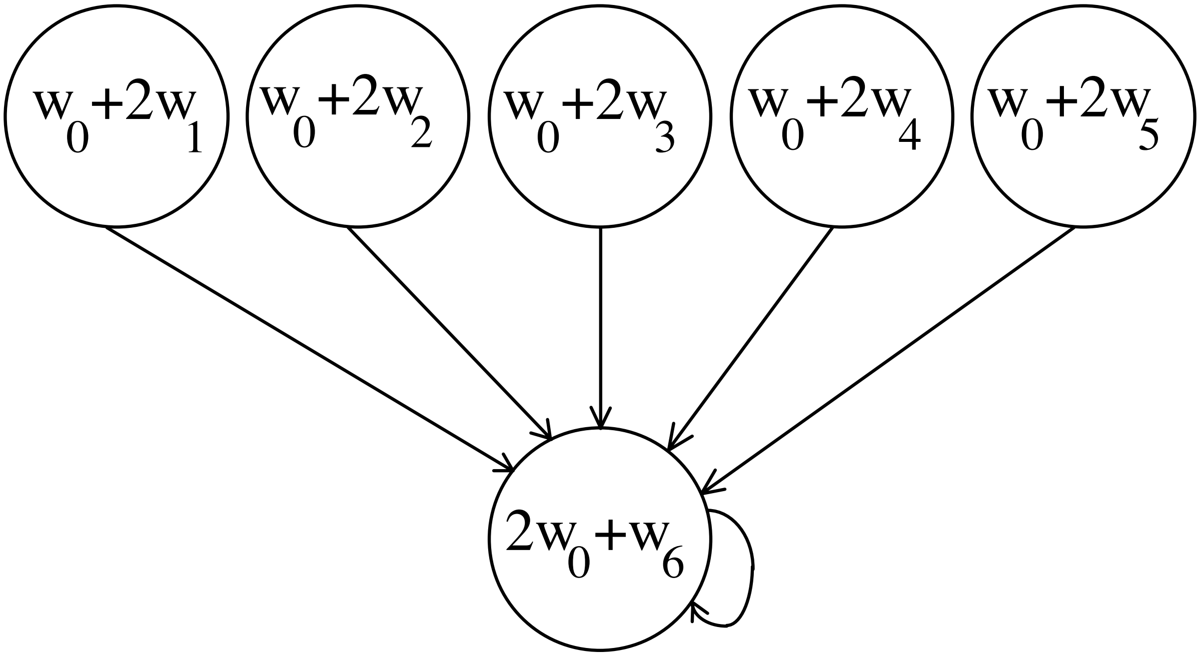
Function approximation system is linear:

- Value is dot product of weight and state vectors:

State 1:	1	2	0	0	0	0	0
State 2:	1	0	2	0	0	0	0
State 3:	1	0	0	2	0	0	0
State 4:	1	0	0	0	2	0	0
State 5:	1	0	0	0	0	2	0
State 6:	2	0	0	0	0	0	1

- State vectors are linearly independent

- State vectors have same magnitude ($1, 2, \infty$ norms)



Gradient descent on mean squared error:

- Define mean squared Bellman residual:

$$E = \sum \left(R + \gamma v_{new} - v_{old} \right)^2$$

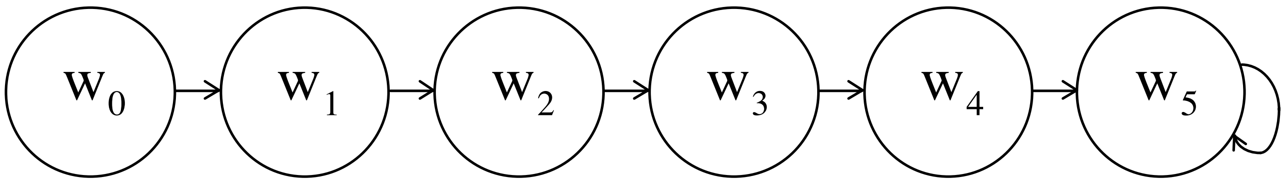
  - Learning equation does gradient descent on *E*:

$$\Delta w = -\alpha \frac{\partial E}{\partial w}$$

  - Guaranteed convergence to a local minimum for epoch-wise.

  - Global minimum if there exists a differentiable mapping from value functions to weight vectors

The Hall Problem:



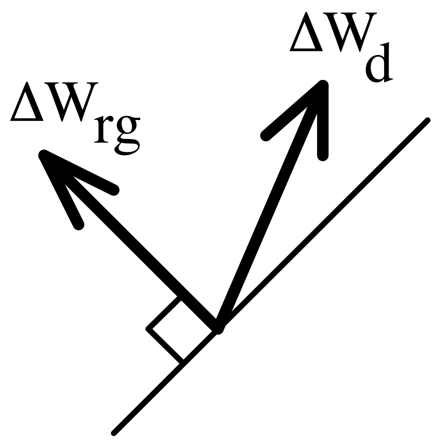
<u>Residual gradient convergence is **very** slow</u>:

- Information flows the wrong direction almost as fast

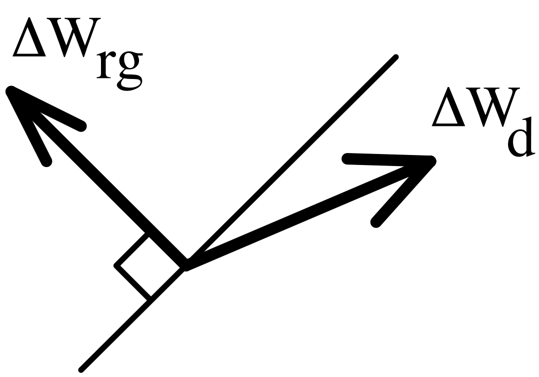
- For 10 states, γ=0.9, mean squared residual is ill conditioned:
    - Hessian eigenvalues differ by ratio of 2000
    - Hessian is not diagonal, eigenvectors at 45° angles
    - Some algorithms ineffective (Delta-bar-delta, quickprop)

- But the direct method is **fast**, and **does** converge!

Direct Method decreases mean squared residual:

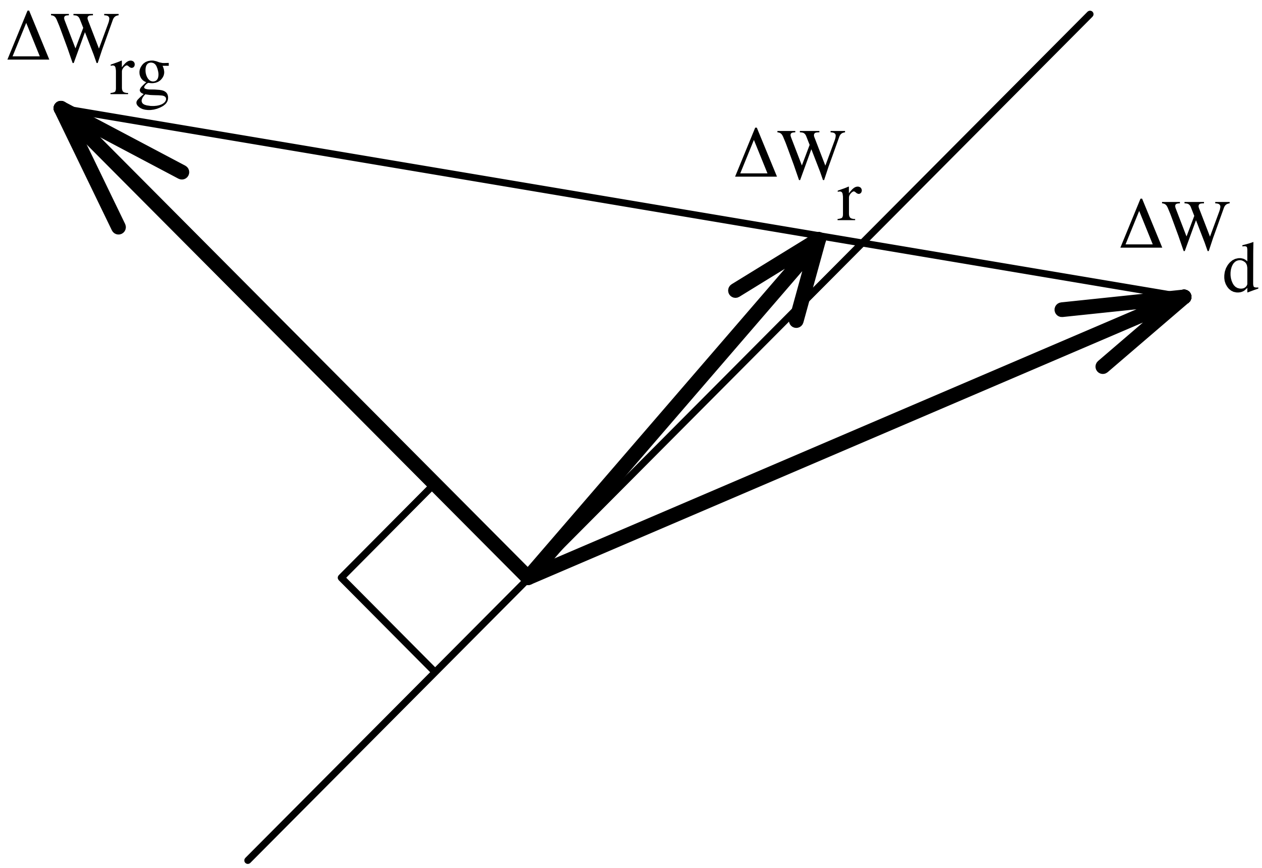


Direct method increases mean squared residual:



- Direct method tends to be fast, if it converges
- Residual gradient converges, but may be slow
- Idea: Find a stable weight change close to direct

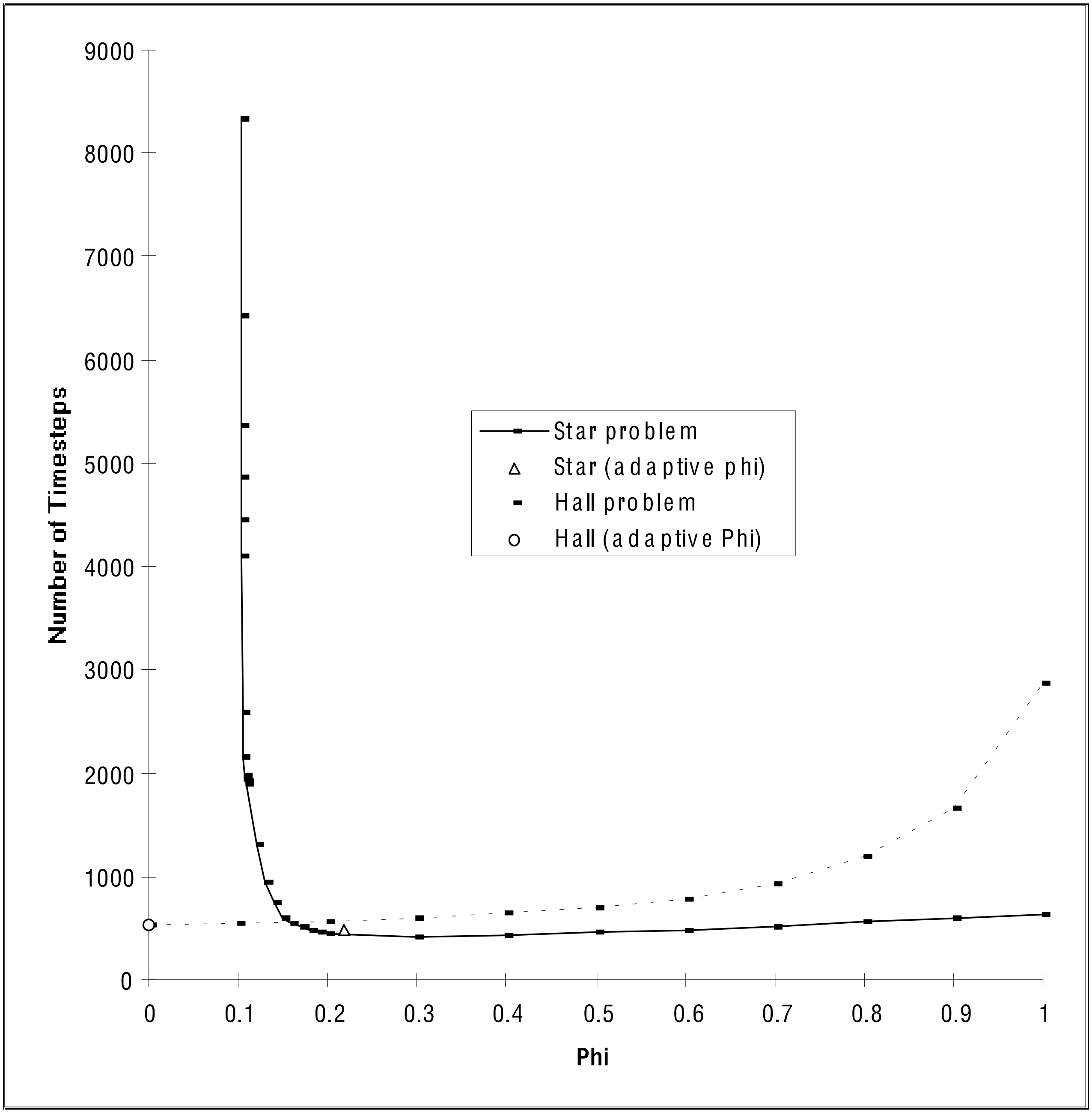
Residual algorithm: linear combination of both:



$$\Delta w_r = \phi \Delta w_{rg} + (1-\phi)\Delta w_d$$

Reinforcement Learning Algorithm	Counterpart of Bellman Equation (top) Weight Change for Residual Algorithm (bottom)
TD(0)	$V(x) = \langle R + \gamma V(x') \rangle$ $\Delta w_r = -\alpha\big(R + \gamma V(x'_1) - V(x)\big)\big(\phi\gamma \tfrac{\partial}{\partial w} V(x'_2) - \tfrac{\partial}{\partial w} V(x)\big)$
Value Iteration	$V(x) = \max_u \langle R + \gamma V(x') \rangle$ $\Delta w_r = -\alpha\big(\max_u \langle R + \gamma V(x') \rangle - V(x)\big)\big(\phi \tfrac{\partial}{\partial w} \max_u \langle R + \gamma V(x') \rangle - \tfrac{\partial}{\partial w} V(x)\big)$
Q-learning	$Q(x,u) = \left\langle R + \gamma \max_{u'} Q(x',u') \right\rangle$ $\Delta w_r = -\alpha\big(R + \gamma \max_{u'} Q(x'_1,u') - Q(x,u)\big)\big(\phi\gamma \tfrac{\partial}{\partial w} \max_{u'} Q(x'_2,u') - \tfrac{\partial}{\partial w} Q(x,u)\big)$
Advantage Learning	$A(x,u) = \left\langle R + \gamma^{\Delta t} \max_{u'} A(x',u') \right\rangle \tfrac{1}{\Delta t} + (1 - \tfrac{1}{\Delta t}) \max_{u'} A(x,u')$ $\Delta w_r = -\alpha\big(\big(R + \gamma^{\Delta t} \max_{u'} A(x'_1,u')\big)\tfrac{1}{\Delta t} + (1 - \tfrac{1}{\Delta t}) \max_{u'} A(x,u') - A(x,u)\big)$ $\cdot\big(\phi\gamma^{\Delta t} \tfrac{\partial}{\partial w} \max_{u'} A(x'_2,u') \tfrac{1}{\Delta t} + \phi(1 - \tfrac{1}{\Delta t}) \tfrac{\partial}{\partial w} \max_{u'} A(x,u') - \tfrac{\partial}{\partial w} A(x,u)\big)$

- Residual algorithms almost identical to direct
- Theoretically should be better
- Mance Harmon found them better in practice

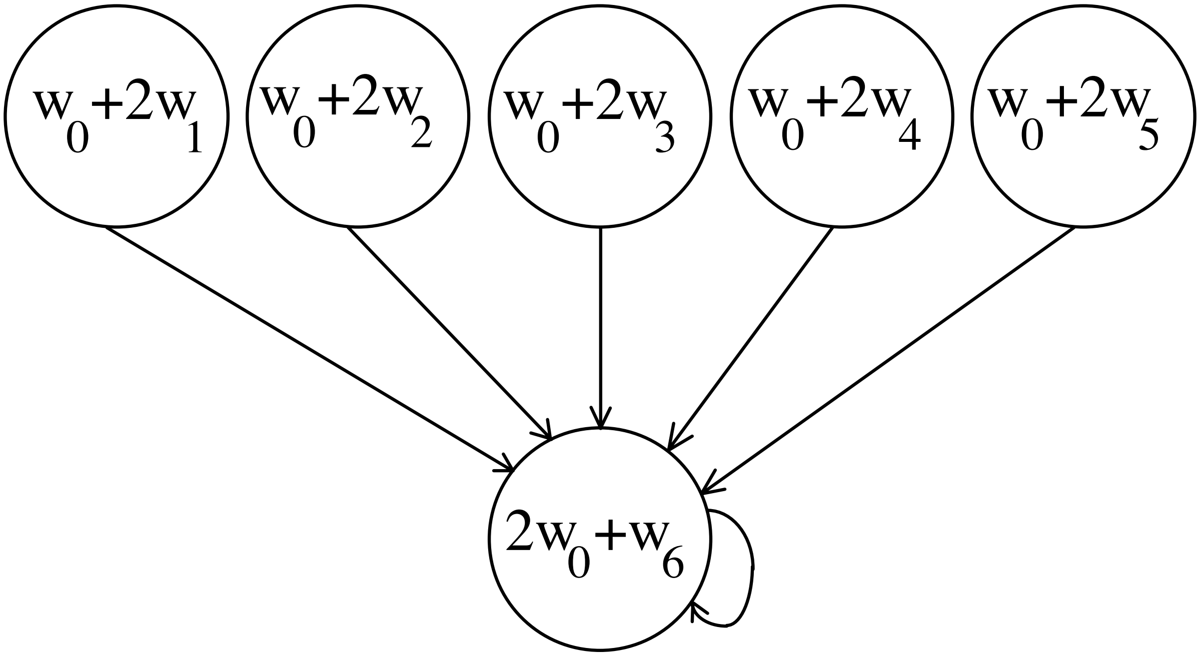


9000
8000
7000
6000
5000
4000
3000
2000
1000
0
Number of Timesteps
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Phi
Star problem
Star (adaptive phi)
Hall problem
Hall (adaptive Phi)

Function Approximation:
Guaranteed Convergence and Convergence Speed

Value Function Approximation Workshop
Machine Learning Conference
July 1995

Leemon Baird
U.S. Air Force Academy
baird@cs.usafa.af.mil
http://kirk.usafa.af.mil/~baird

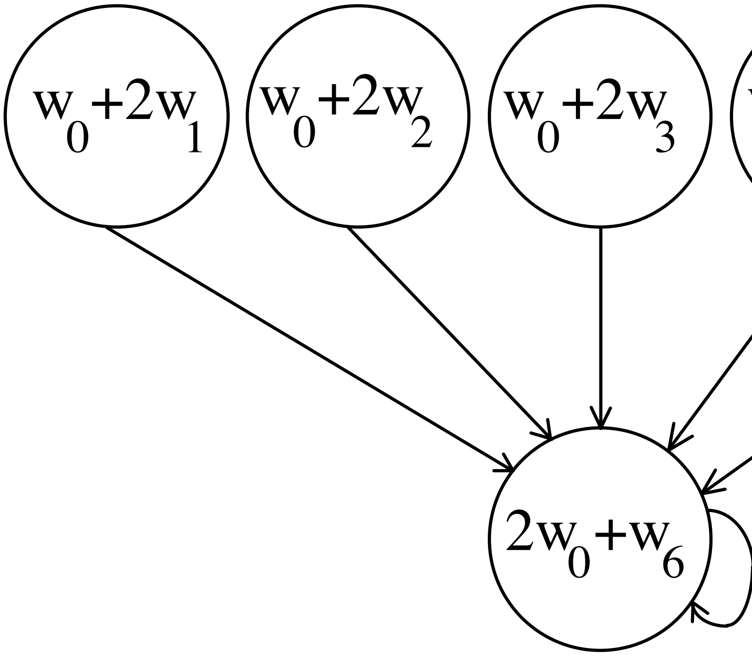


Function approximation system is linear:

- Value is dot product of weight and state vectors:

State 1:	1	2	0	0	0	0	0
State 2:	1	0	2	0	0	0	0
State 3:	1	0	0	2	0	0	0
State 4:	1	0	0	0	2	0	0
State 5:	1	0	0	0	0	2	0
State 6:	2	0	0	0	0	0	1

- State vectors are linearly independent
    - State vectors have same magnitude (1, 2, ∞ norms)



Gradient descent on mean squared error:

- Define mean squared Bellman residual:

$$E = \sum \left(R + \gamma v_{new} - v_{old} \right)^2$$

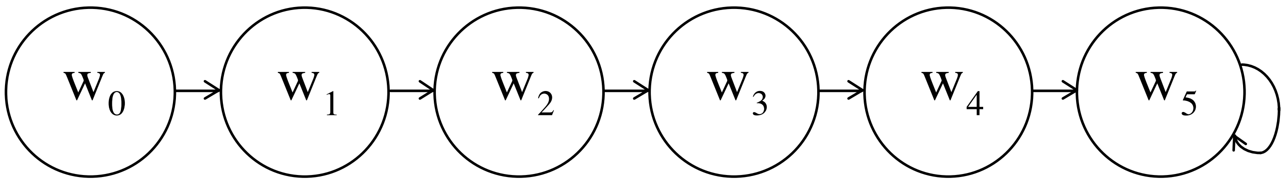
  - Learning equation does gradient descent on *E*:

$$\Delta w = -\alpha \frac{\partial E}{\partial w}$$

  - Guaranteed convergence to a local minimum for epoch-wise.

  - Global minimum if there exists a differentiable mapping from value functions to weight vectors

The Hall Problem:

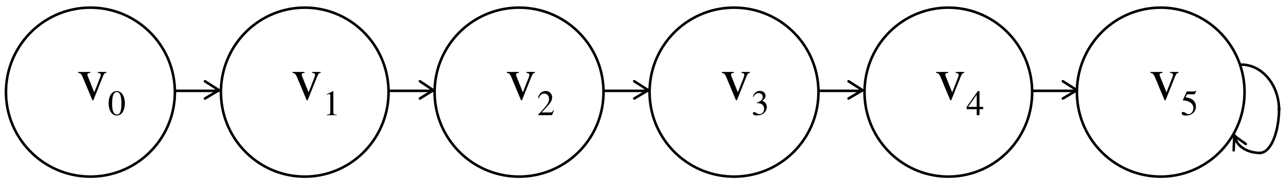


<u>Residual gradient convergence is **very** slow</u>:

- Information flows the wrong direction almost as fast

- For 10 states, γ=0.9, mean squared residual is ill conditioned:
    - Hessian eigenvalues differ by ratio of 2000
    - Hessian is not diagonal, eigenvectors at 45° angles
    - Some algorithms ineffective (Delta-bar-delta, quickprop)

Hand craft state vectors based on known model:



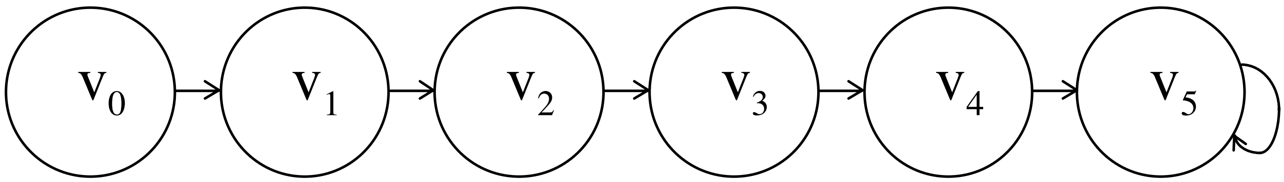
<u>Ensure each weight controls one difference:</u>

- Value is dot product of weight and state vectors:

State 0:	1	1	1	1	1	1
State 1:	1	1	1	1	1	0
State 2:	1	1	1	1	0	0
State 3:	1	1	1	0	0	0
State 4:	1	1	0	0	0	0
State 5:	1	0	0	0	0	0

- For 10 states, eigenvalue ratio decreases from 2000 to 20

Prior knowledge of topology, not order:



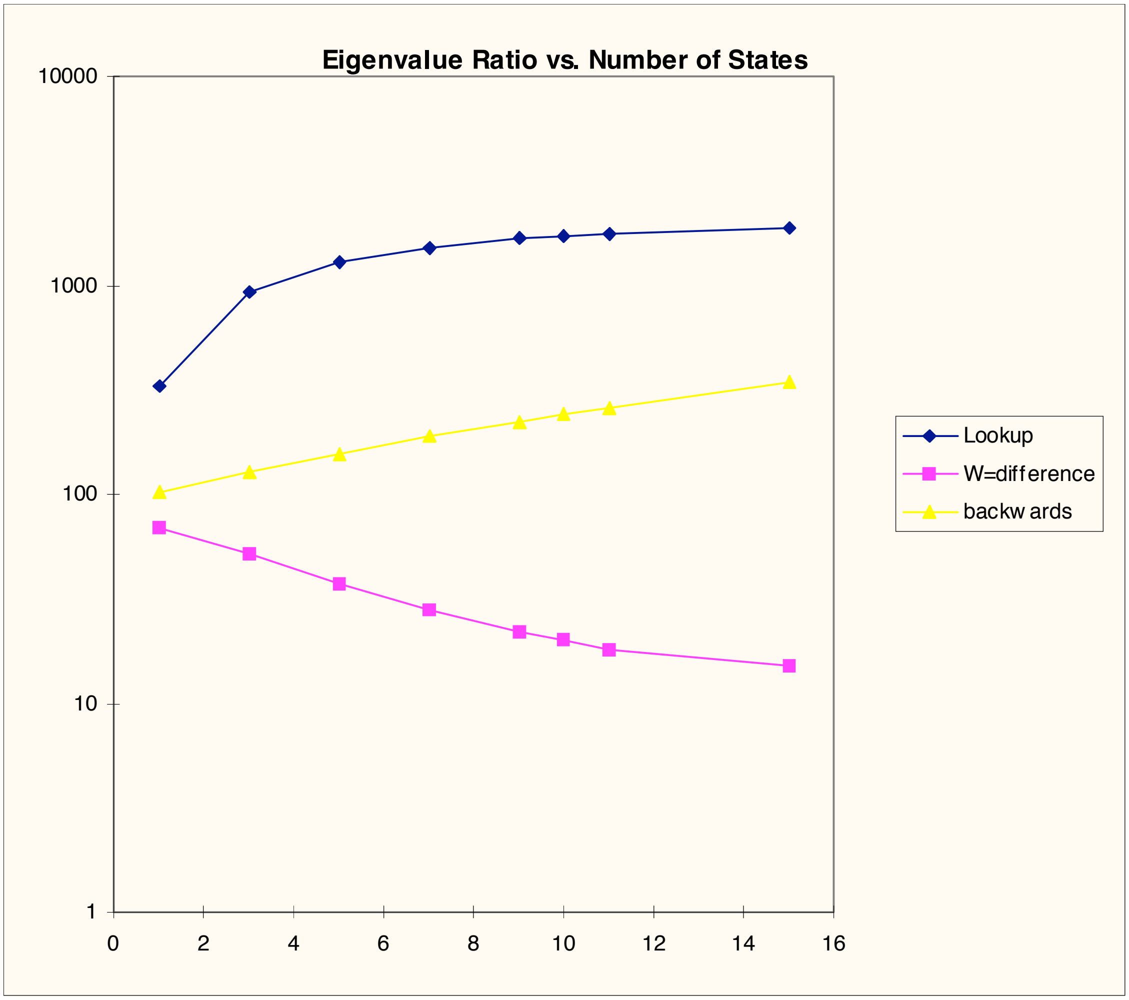
Slight bias to generalize the wrong direction:

- Value is dot product of weight and state vectors:

State 0:	1	0	0	0	0	0
State 1:	1	1	0	0	0	0
State 2:	1	1	1	0	0	0
State 3:	1	1	1	1	0	0
State 4:	1	1	1	1	1	0
State 5:	1	1	1	1	1	1

- For 10 states, eigenvalue ratio increases from 20 to 200

How conditioning changes with number of states



-Longer halls are even worse for 2 systems

- Longer halls are better with all prior info

    -- Still levels out at ratio of 10

    -- Still impractically slow

Summary:

- Direct method can blow up on simple problems

- Impractical to hand craft fast function approximation systems
    - Goal: develop an algorithm that:
    -- Works with any function approximator
    -- Guarantees convergence like residual gradient
    -- Is as fast as the direct method

- Goal theoretically met by Residual algorithms

- Mance Harmon showed it works in practice