# Recurrent Networks

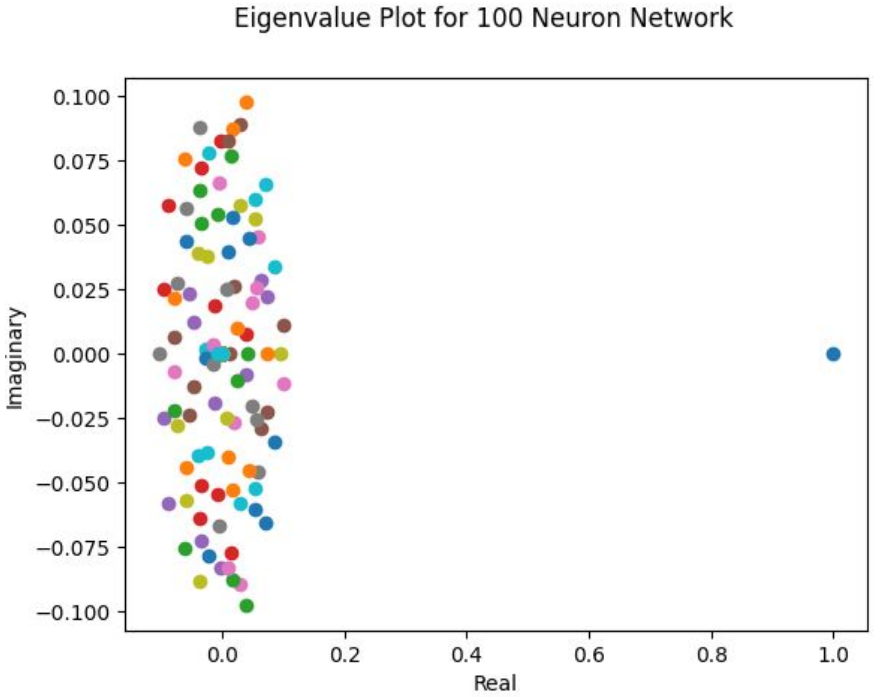
By Alexandru Tapus

# Presentation Outline

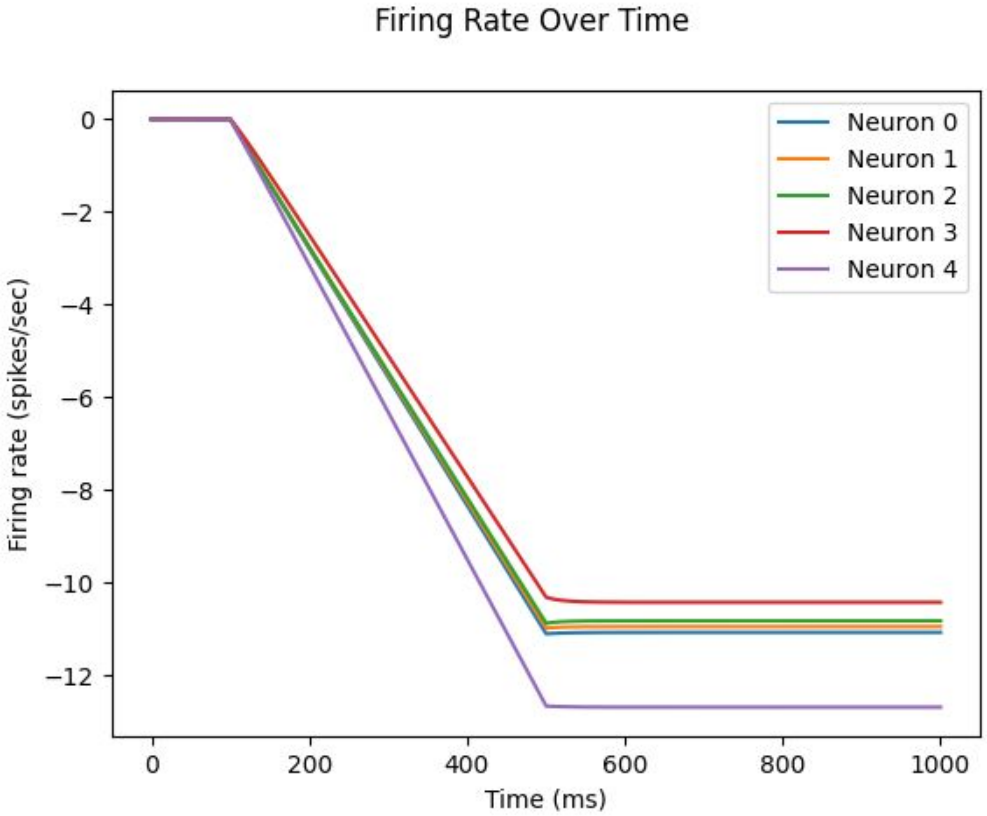
- Review linear networks and their requirements for integrating input.
- Discuss the methods and training curves for modeling eye position maintenance using nonlinear networks.
- Outline how the weight matrix was fit to achieve the task.
- Show several graphs of the network running simulations.

# Linear Recurrent Integrators

# Random Linear Recurrent Network

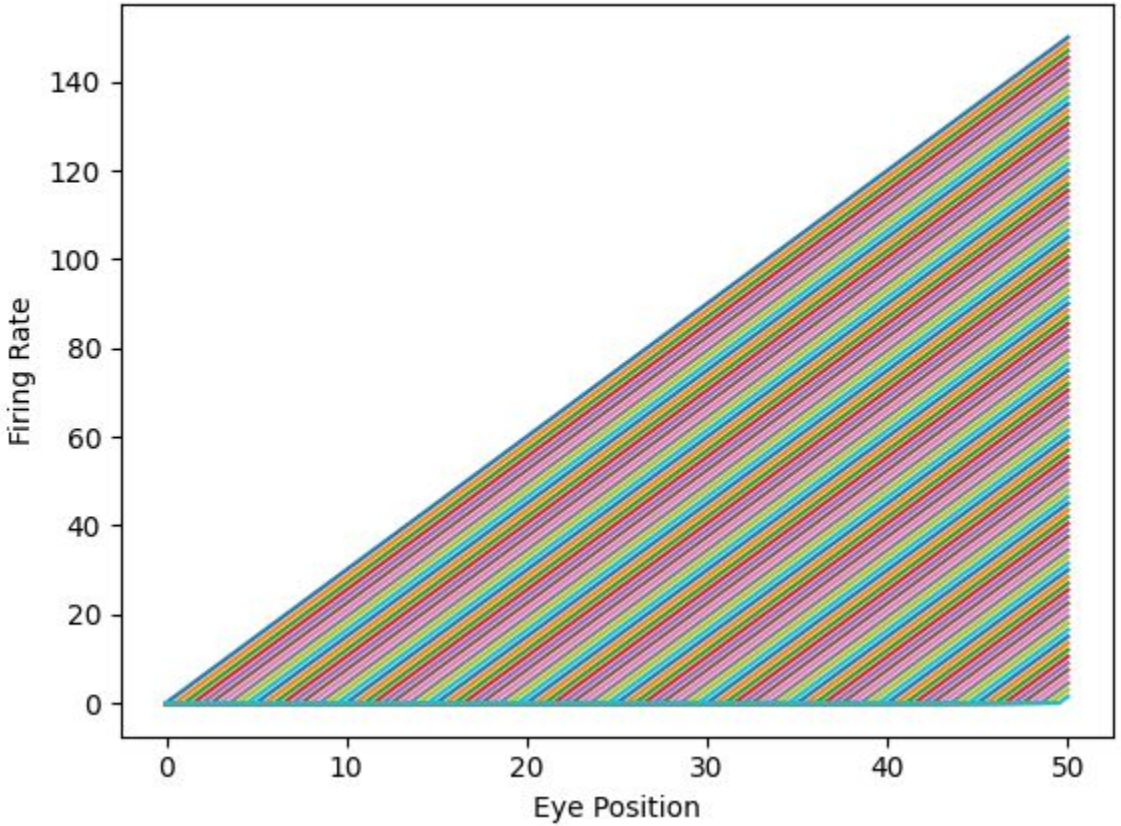


# Applying Current Along an Eigenvector



# Nonlinear Network for Maintaining Eye Position

# Target Fixed Points Over Eye Position



# Equations of the Network

Each neuron is modeled by the following equation:

- $\tau \, dr/dt = -r + WS(r) + T + I(t)$

Where

- $S(r) = r^p/(a+r^p)$
  - $p=1.4$, $a=.4$
  - These exact values aren't necessary for the fit.

# Fitting the Weight Matrix

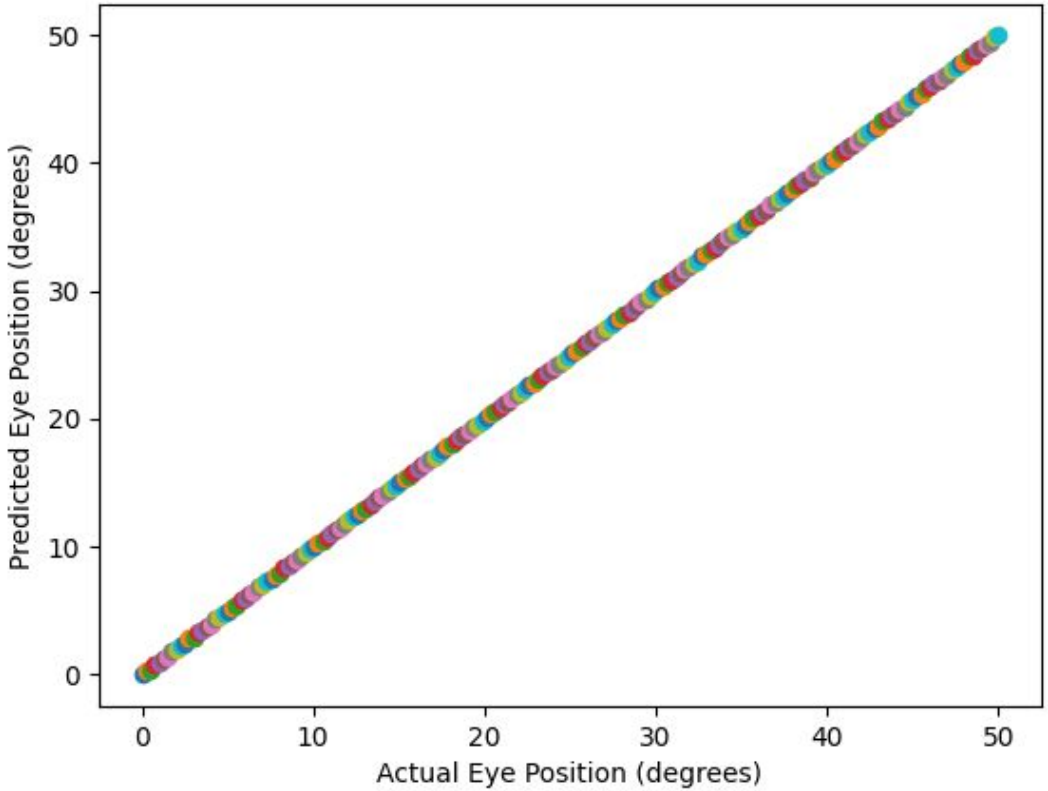
- $\tau\, dr/dt = -r + \mathbf{W}s(r) + T + I(t)$ SO $r = \mathbf{W}s(r) + T$
  - Assumes no input when looking at fixed points
- $\mathbf{W} = \xi \times \eta^T$
  - $\xi$ is set to (3 …3) of the same length as the number of neurons
  - We fit for η by using linear regression
- Define $\hat{E} = \sum S(r_e) * \eta_e = E$ FIT $E = \mathbf{S(r)} * \eta$
  - **S(r)** is a matrix exn holding values at each eye position for each neuron
- Set the weight matrix as the cross product $\mathbf{W} = \xi \times \eta$

# Predicting Eye Position

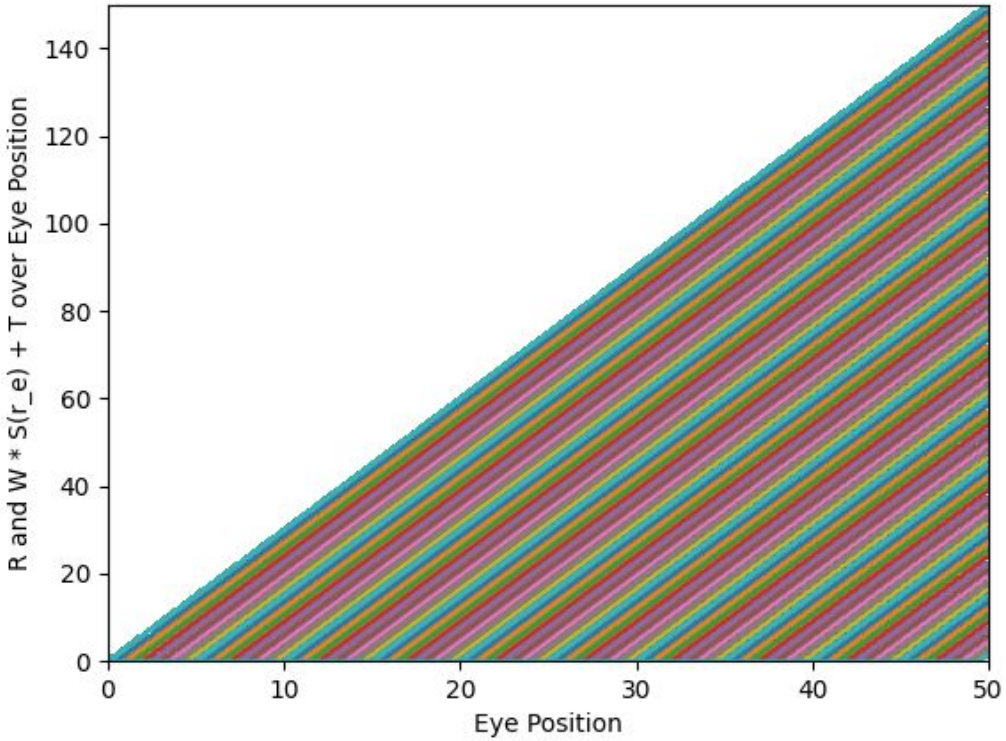
For predicting eye position we use the previous formula:

- $\hat{E} = \sum S(r_e) * \eta_e$

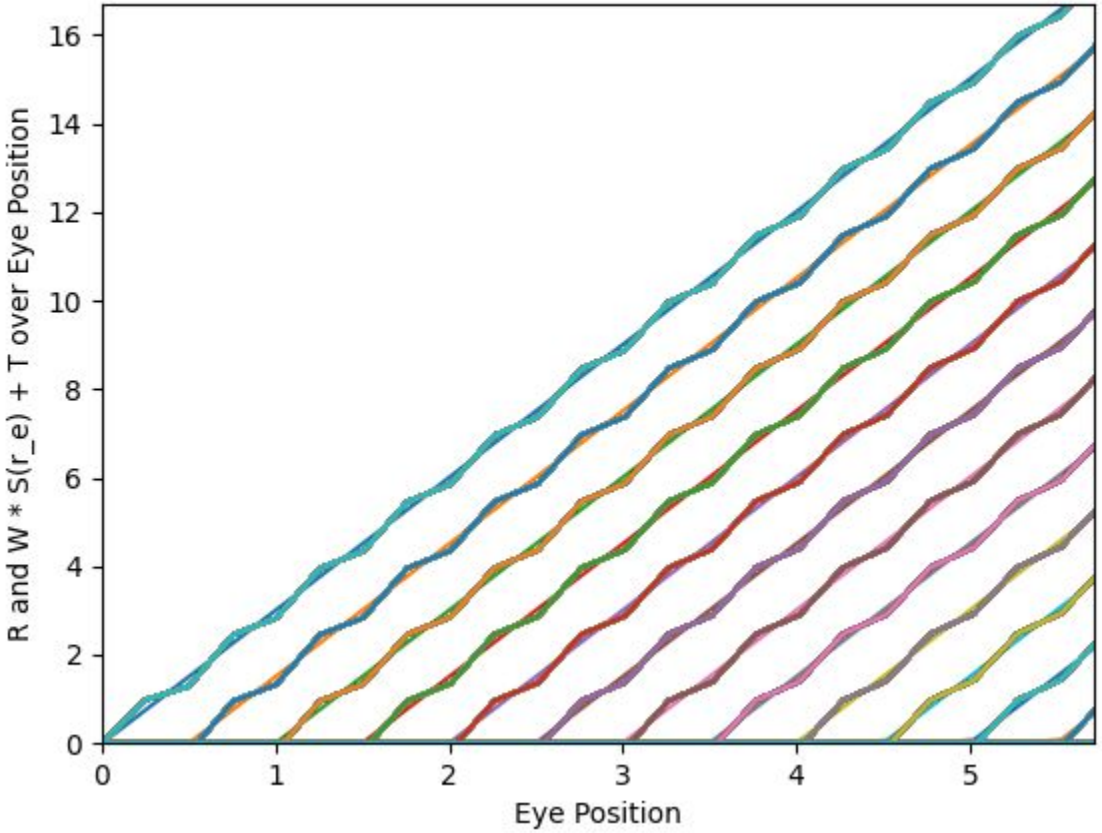
# Predicting Eye Position Results



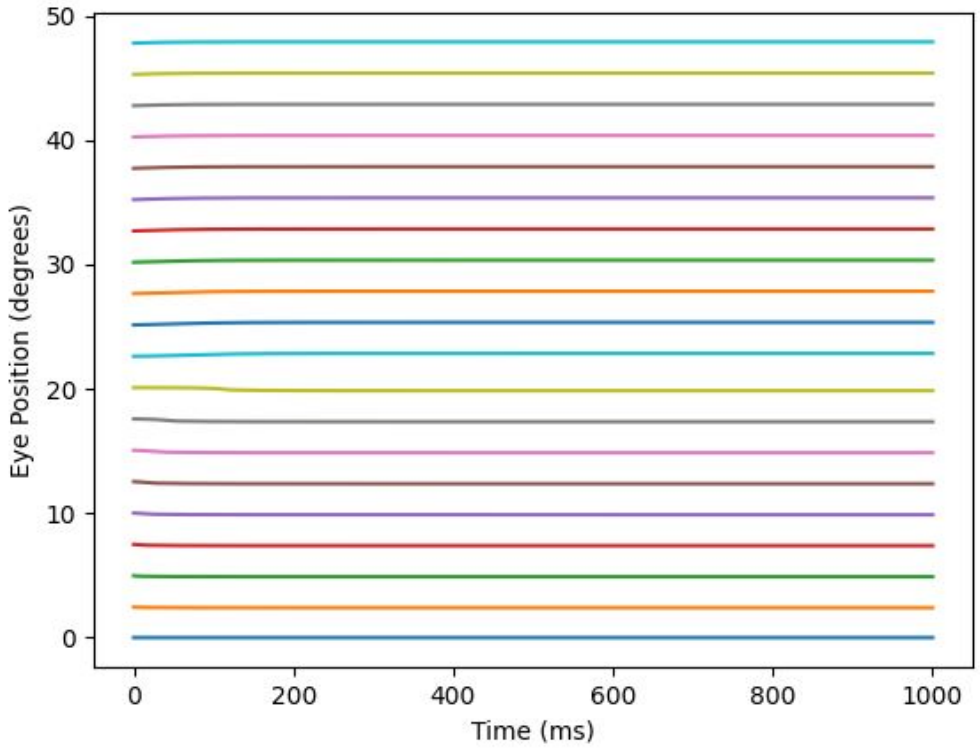
# Visualizing the Achieved Fixed Points



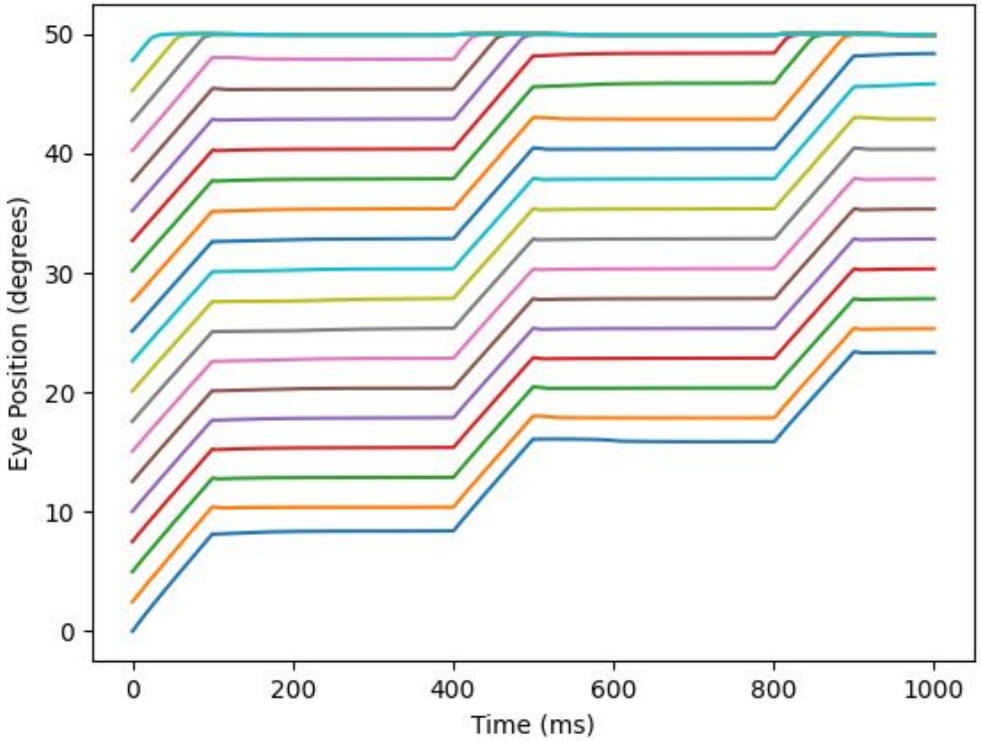
# Visualizing the Achieved Fixed Points



# Maintaining a Preset Eye Position



# Simulating an Eye Movement



# Conclusion

- Using identical rows for the weight matrix allows the network to fit many fixed points along the target curves.
- $\eta$ and $S(r)$ at each eye position can be used to get an accurate prediction of eye position from firing rates.
- The network integrates external input that is equal for all neurons.

# Future Work

- Training the network with no assumptions on the weights.
- Creating two separate populations that model positive and negative eye positions.
- Comparing the weights arrived at by both the networks.

# Project Files

RecruiterNeuronsAttempt2.py on Github