General Principles of Human and Machine Learning

Lecture 10: Function Learning

Dr. Charley Wu

https://hmc-lab.com/GPHML.html

Welcome back!

Teaching evaluations

- You should have recieved an email asking to submit your teaching evaluations
- Please do so before January 20th

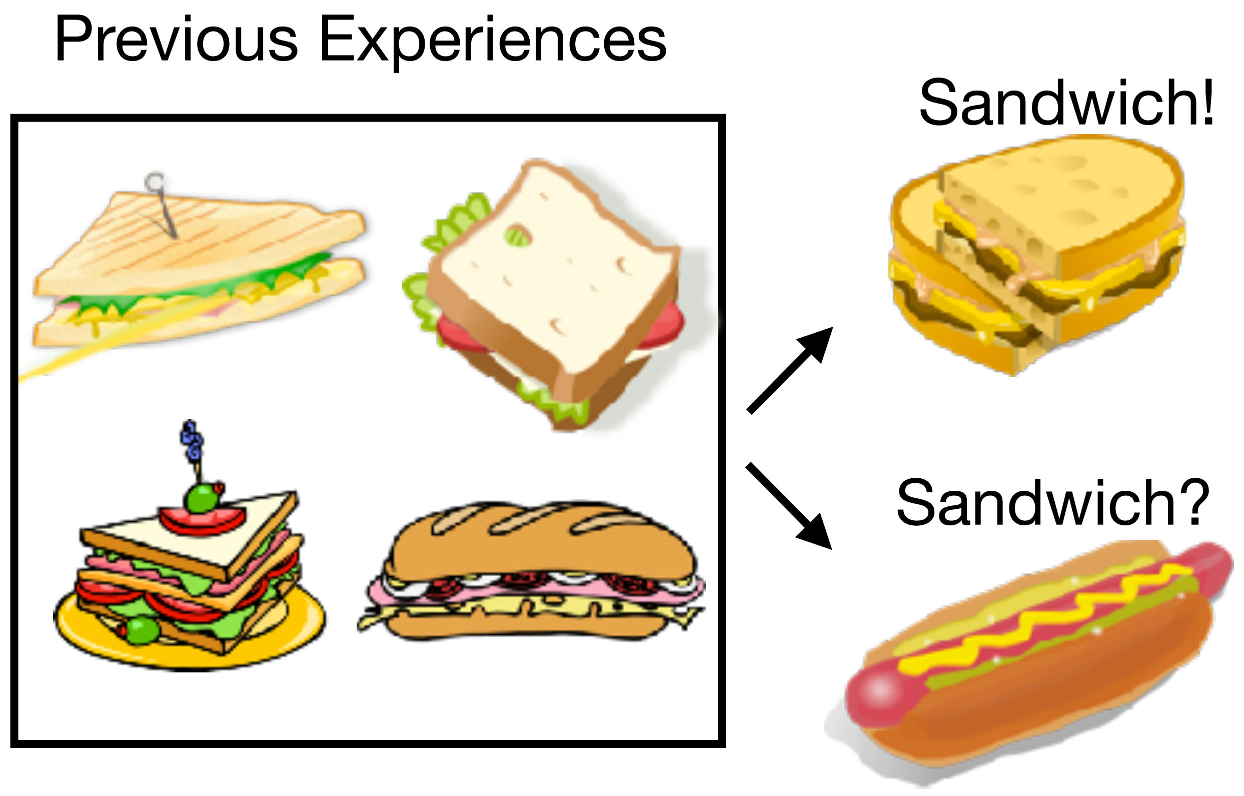
Exam registration

- This should now be “theoretically possible” depending on the “Prüfungsordnung” of your study program
- If you are on the new Prüfungsordnung, you can register on ALMA
- If you are on the old one and unable to register, please let me know at the next lecture and I can get you manually added

Week 10:		Jan 14: Function learning	Jan 15	Alex	Wu, Meder, & Schulz (in press)
Week 11:	→	Jan 21: No Lecture	Jan 22: No Tutorial		
Week 12:		Jan 28: Language and semantics	Jan 29	Hanqi	Kamath et al., (2024)
Week 13:		Feb 4: General Principles	Feb 5	Charley	Gershman (2023)
Exam 1	13:00-15:00 21.02.2025 Hörsaal 1 F119 (SAND)				
Exam 2	12:00-14:00 11.04.2025 Ground floor lecture room, AI building, Maria-von-Linden-Str. 6, D-72076 Tübingen				

The story so far …

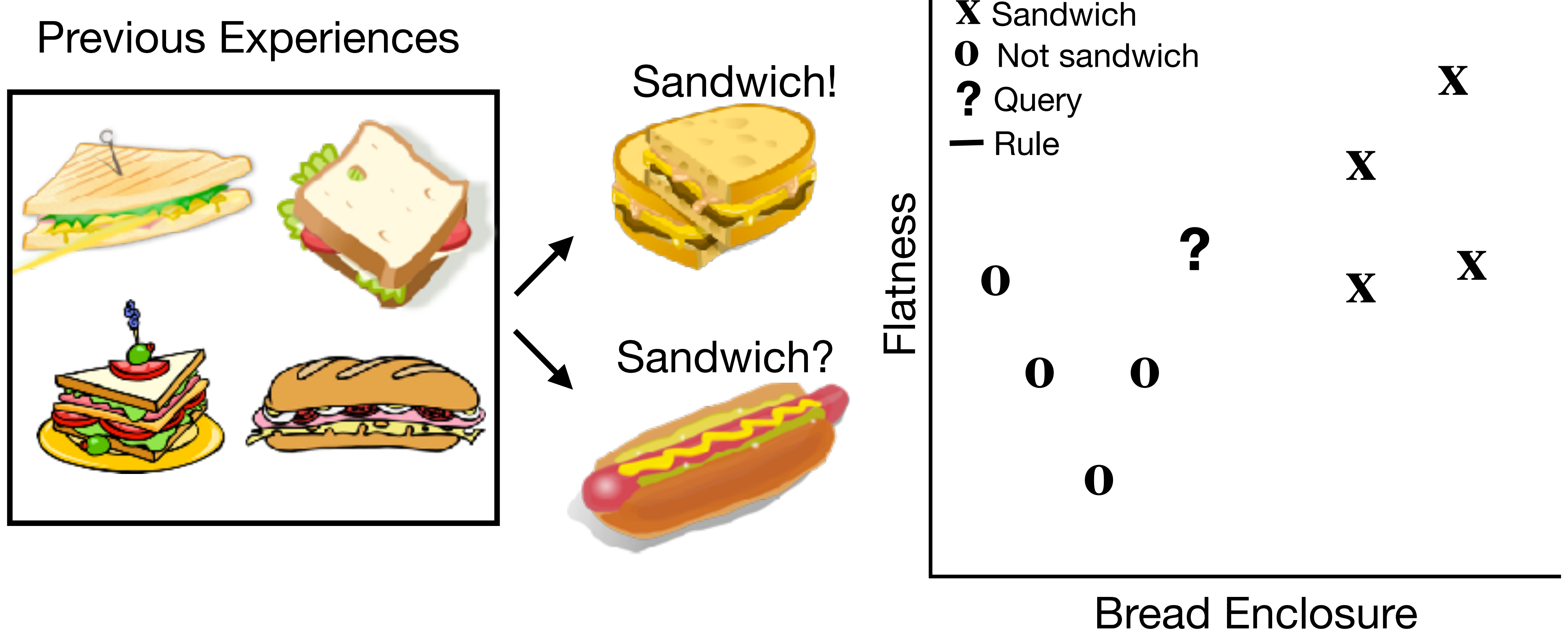
Concept learning as classification



The story so far …

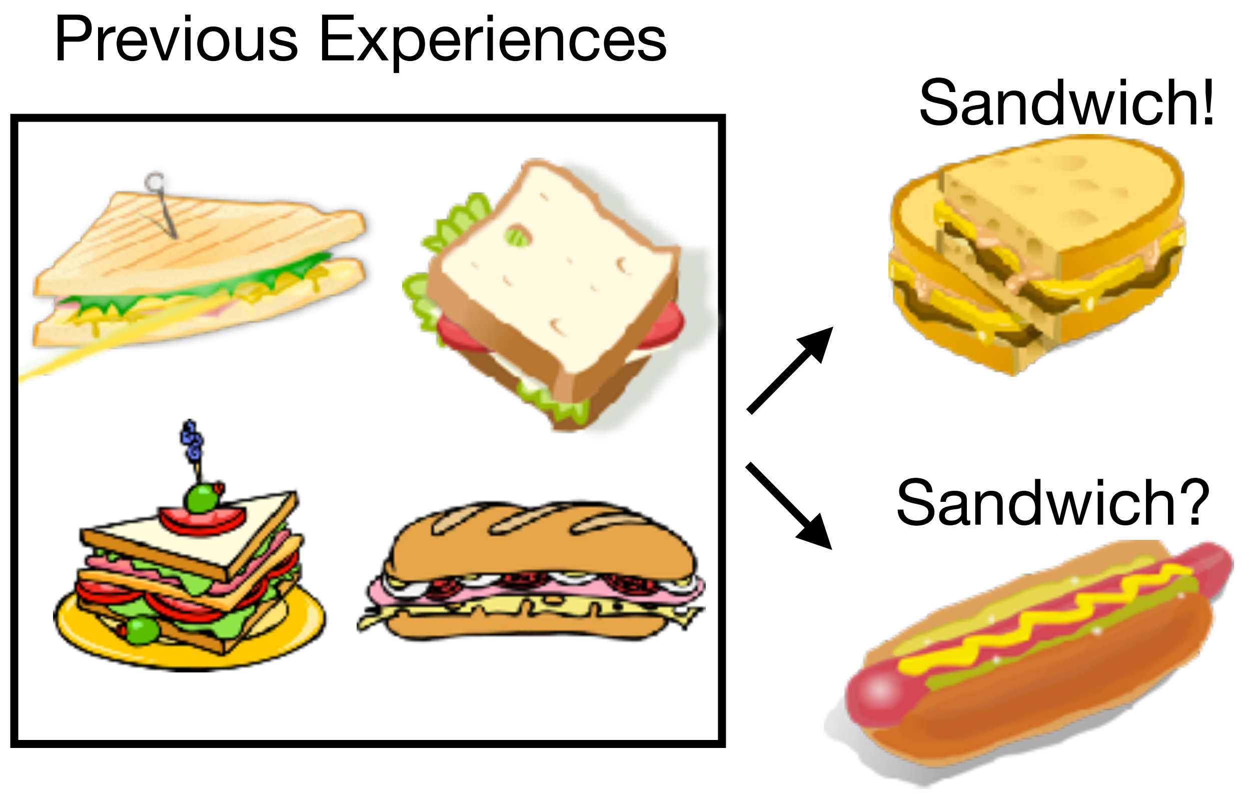
Concept learning as classification

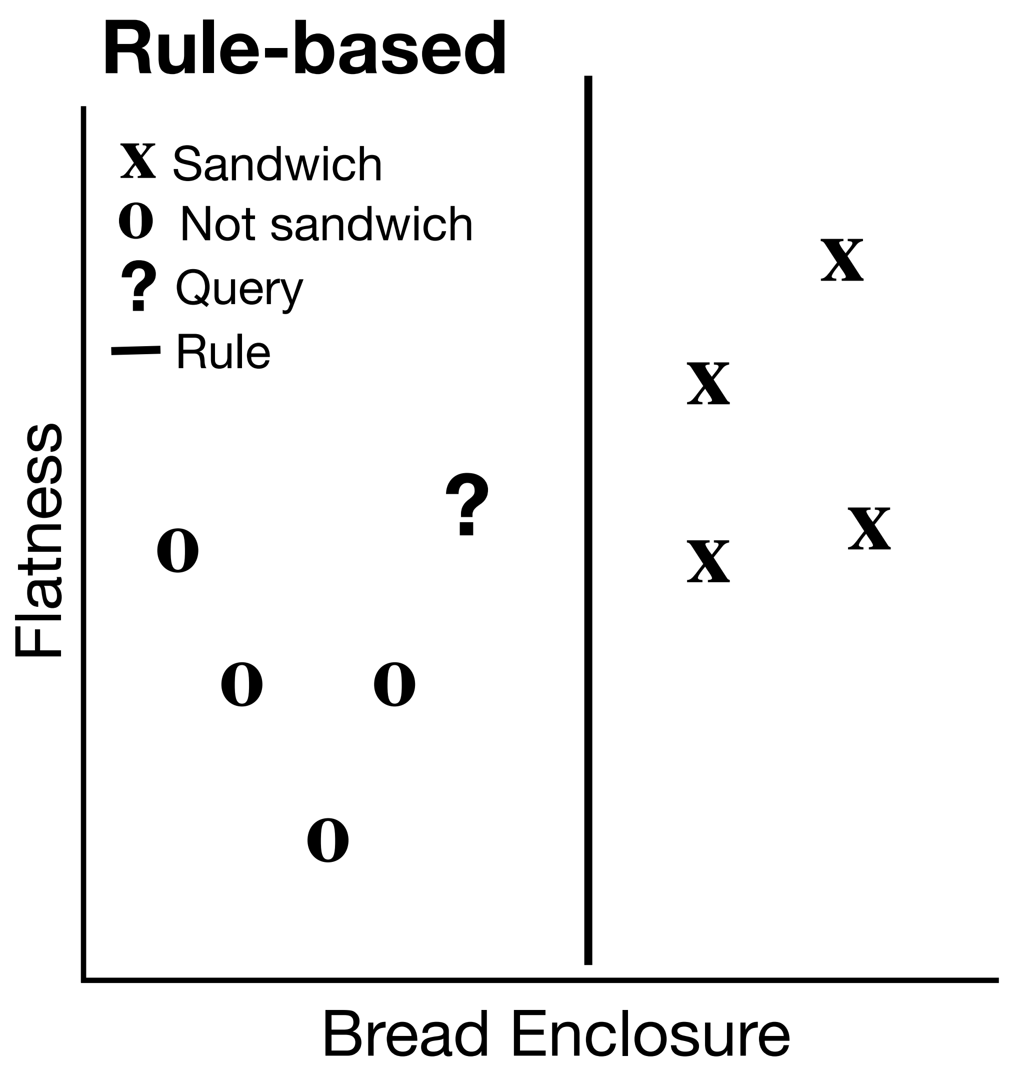
Rule-based



The story so far …

Concept learning as classification

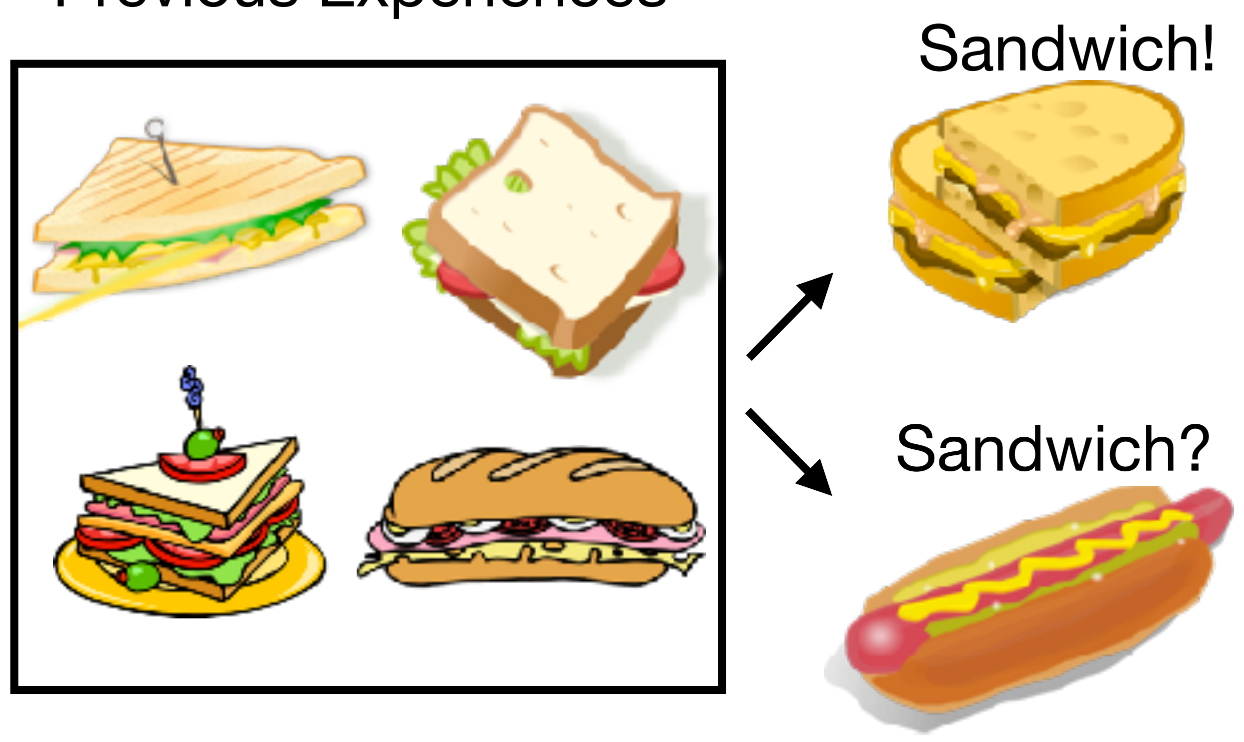




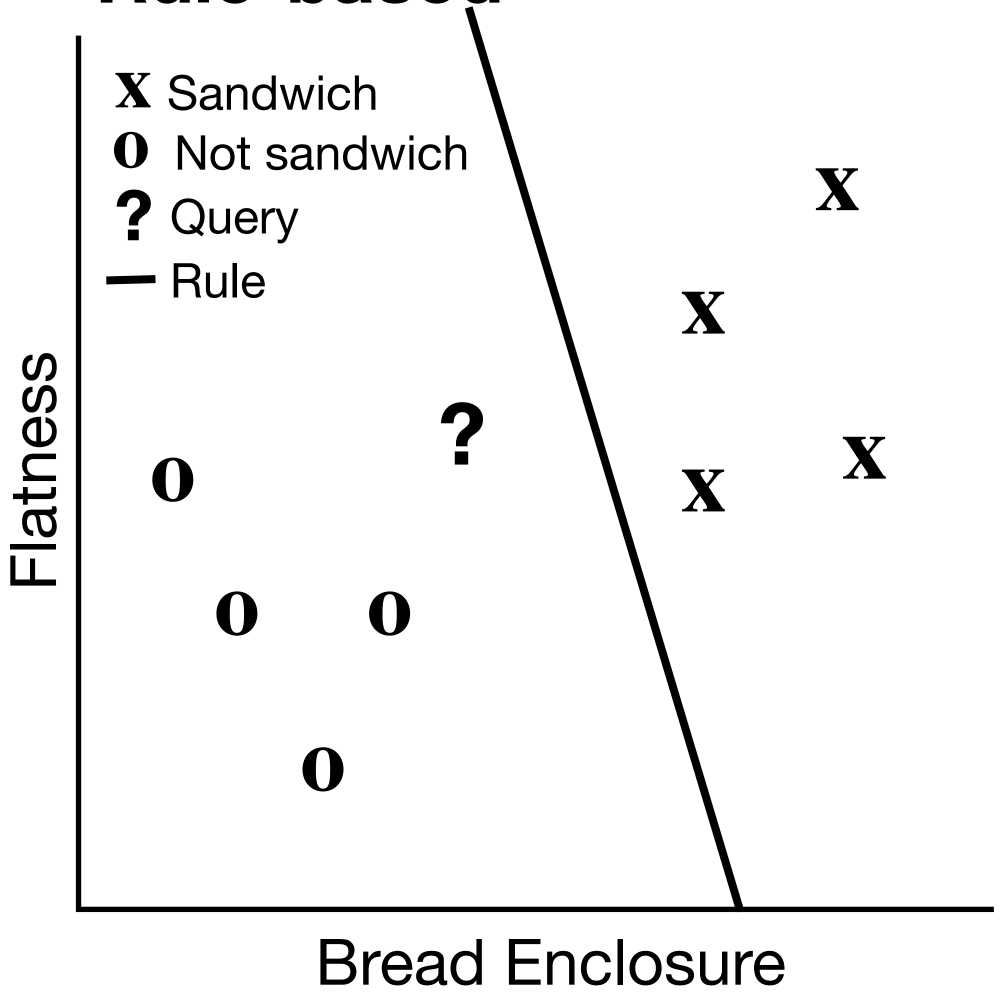
The story so far …

Concept learning as classification

Previous Experiences



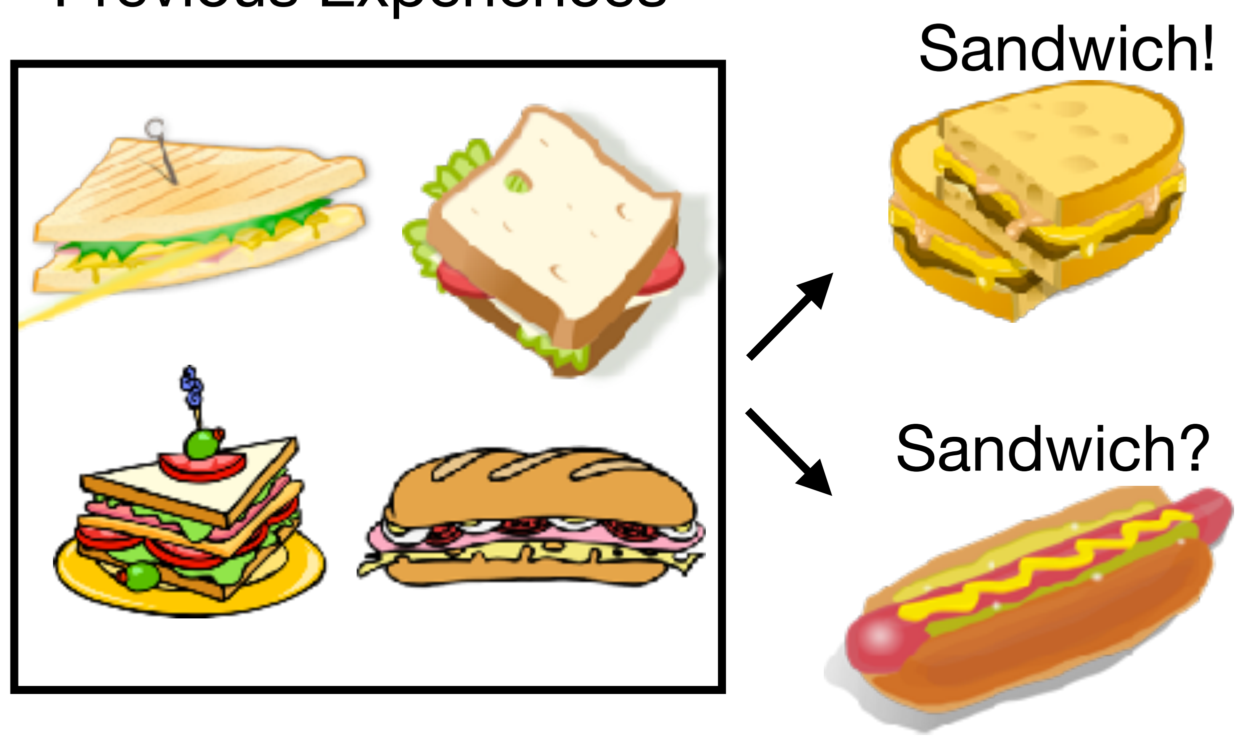
Rule-based



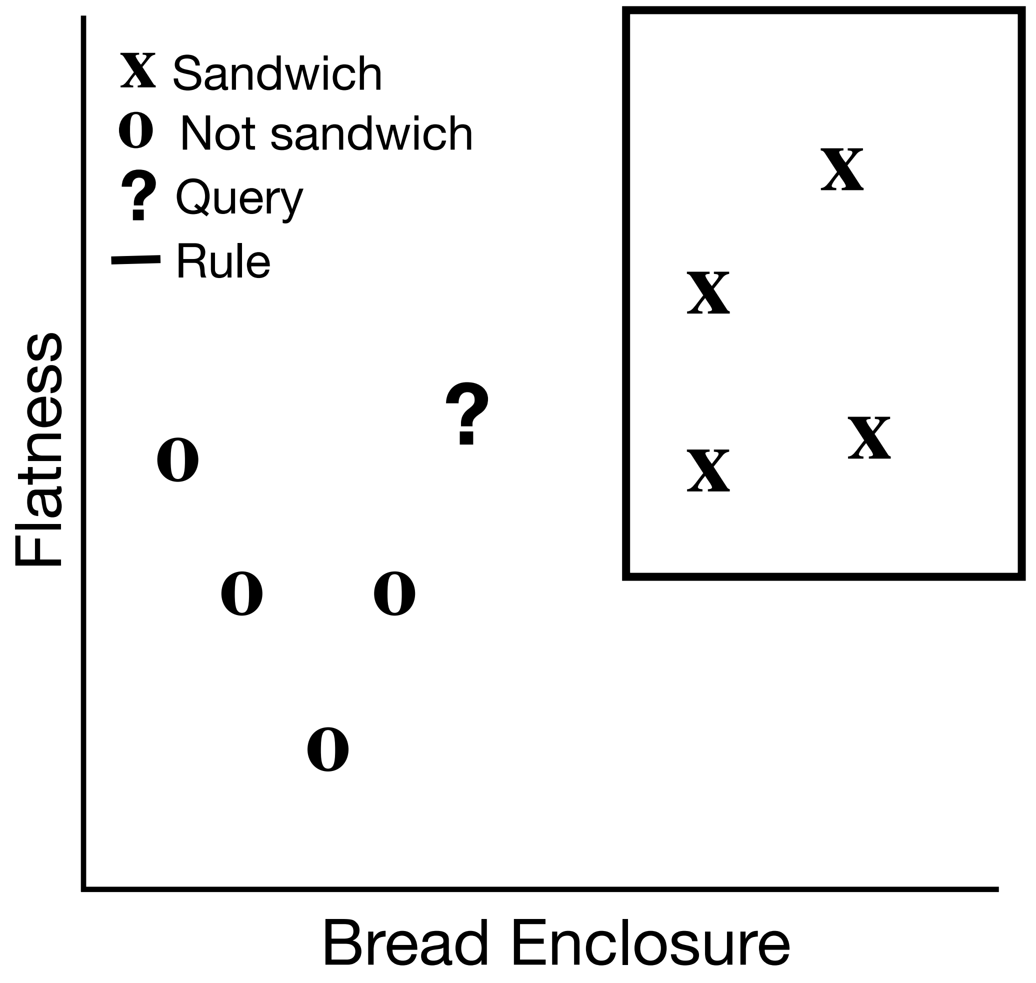
The story so far …

Concept learning as classification

Previous Experiences

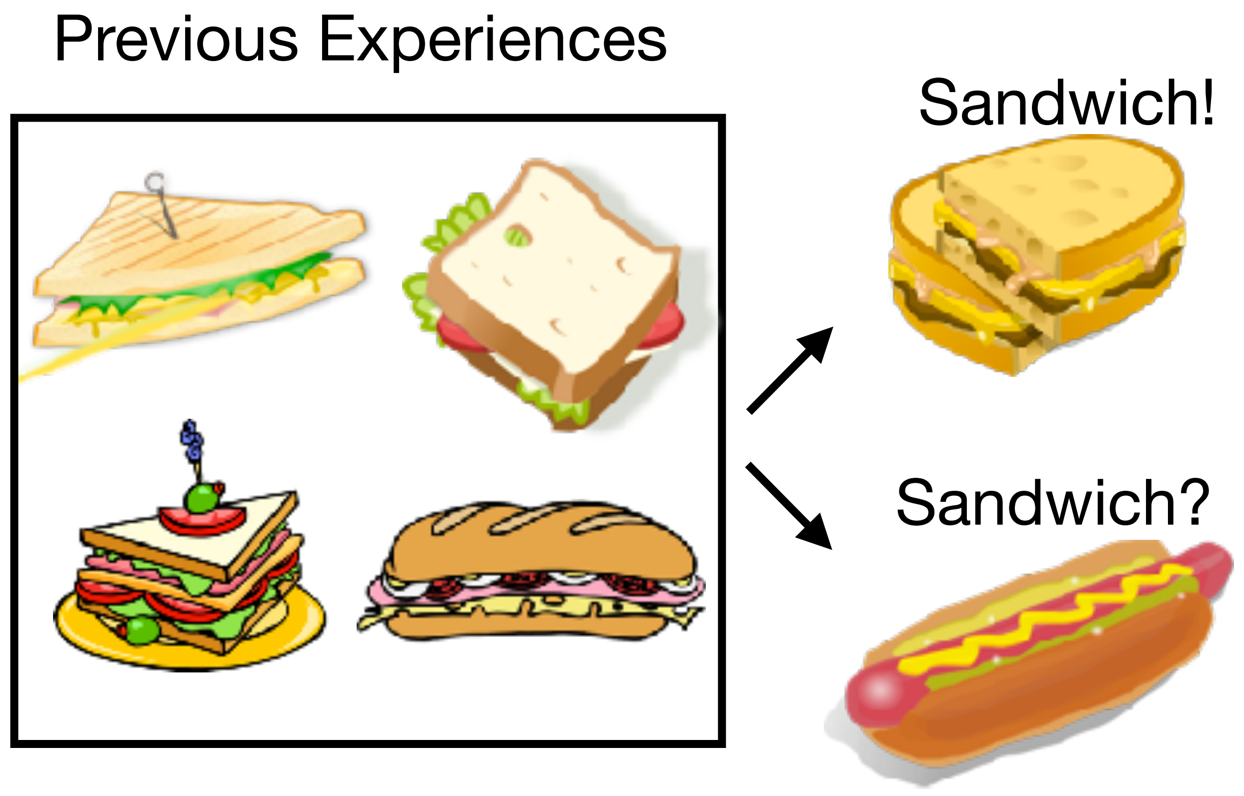


Rule-based

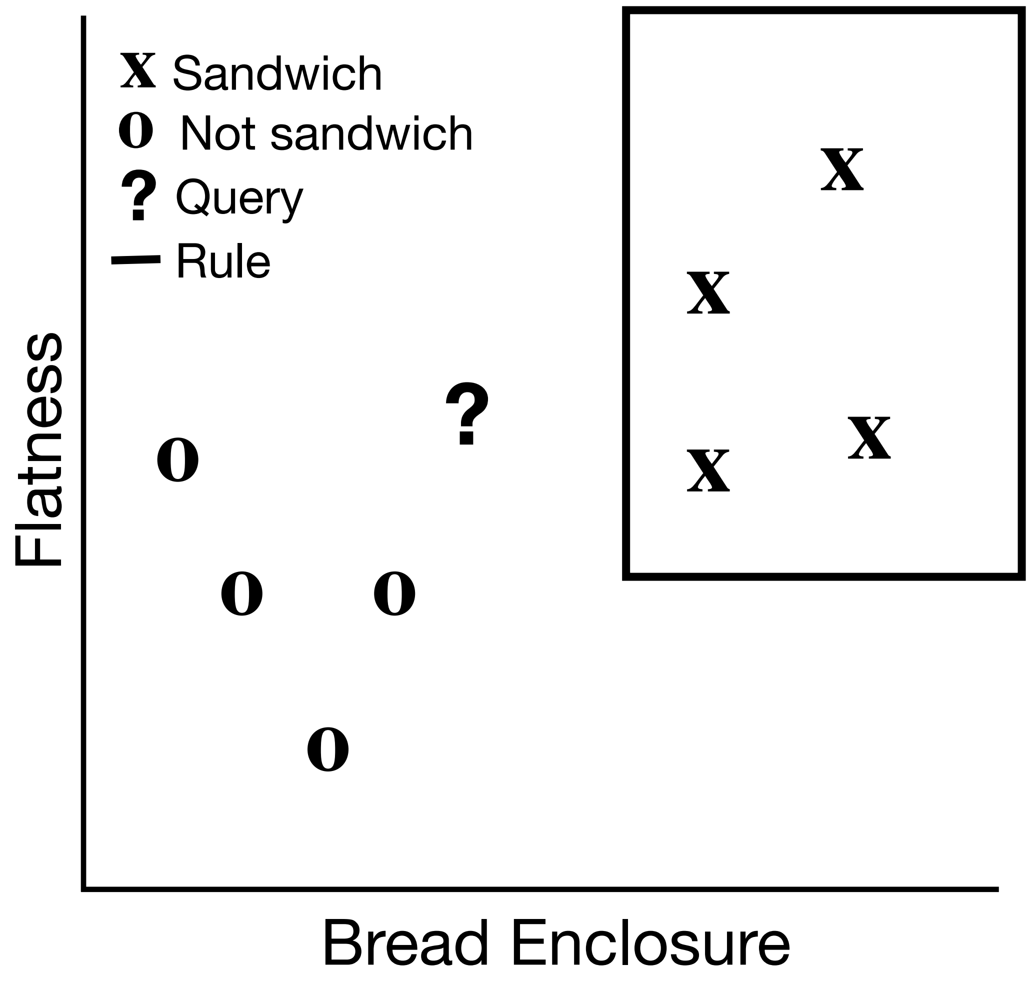


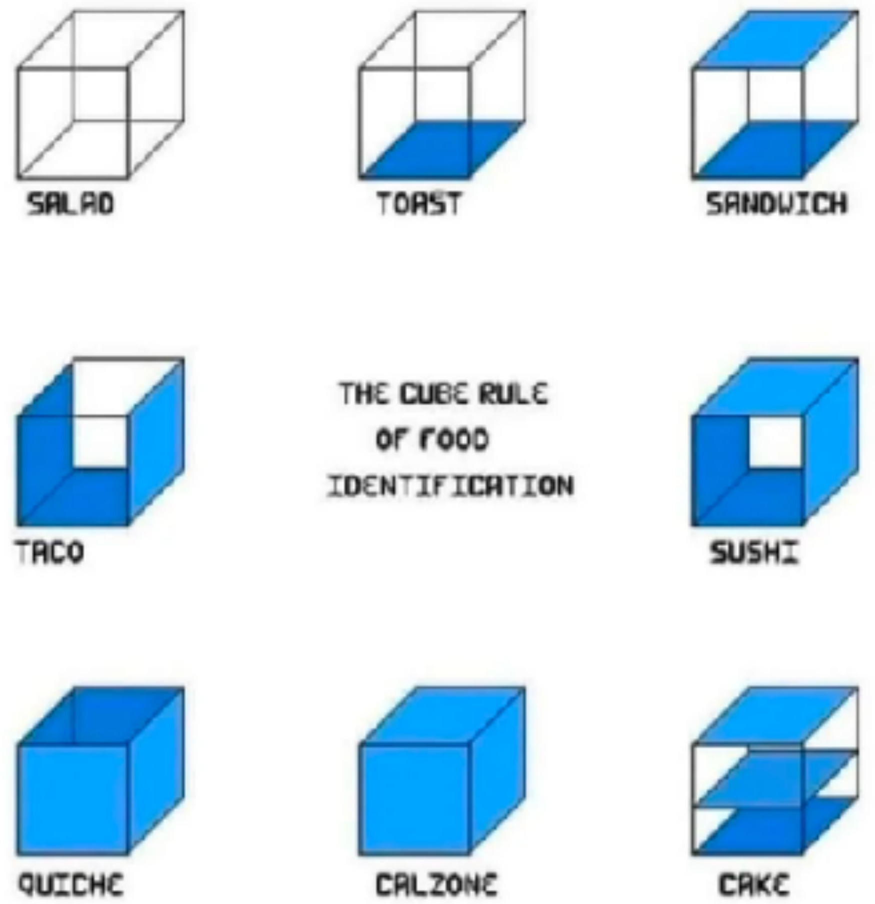
The story so far …

Concept learning as classification



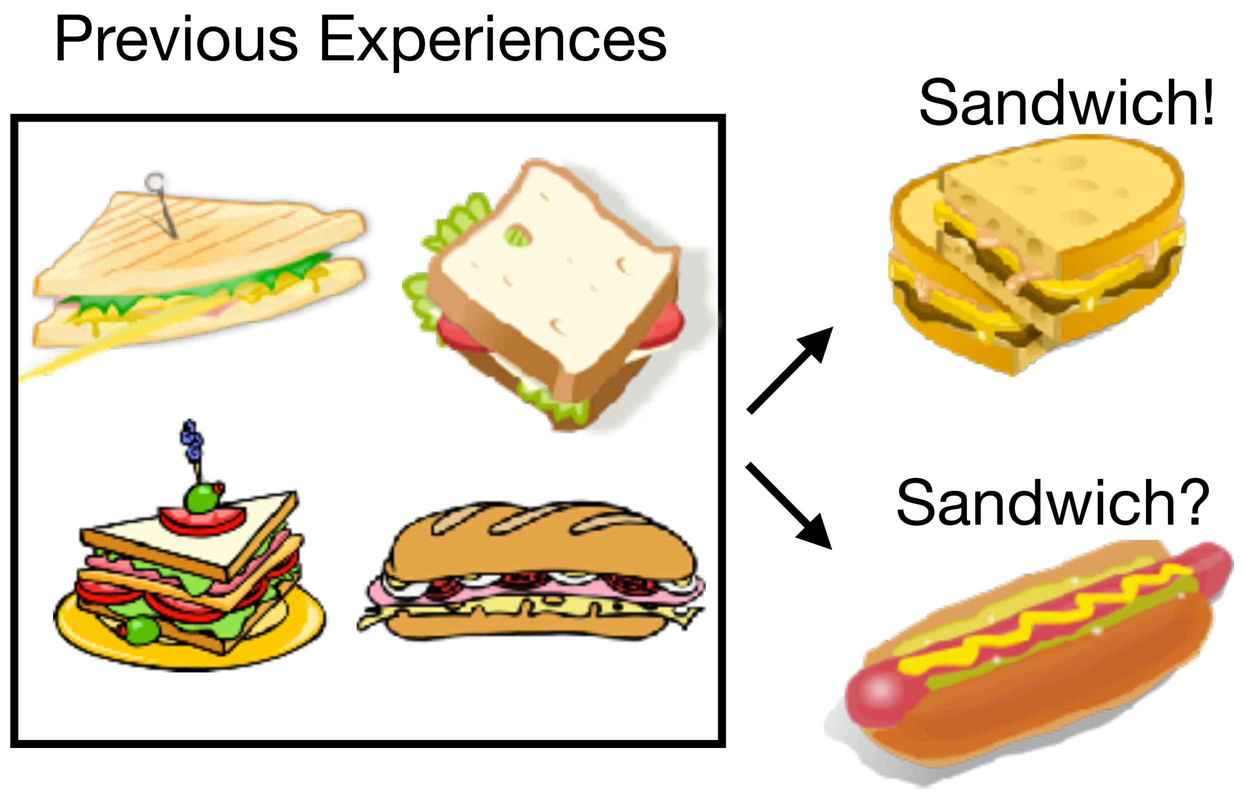
Rule-based



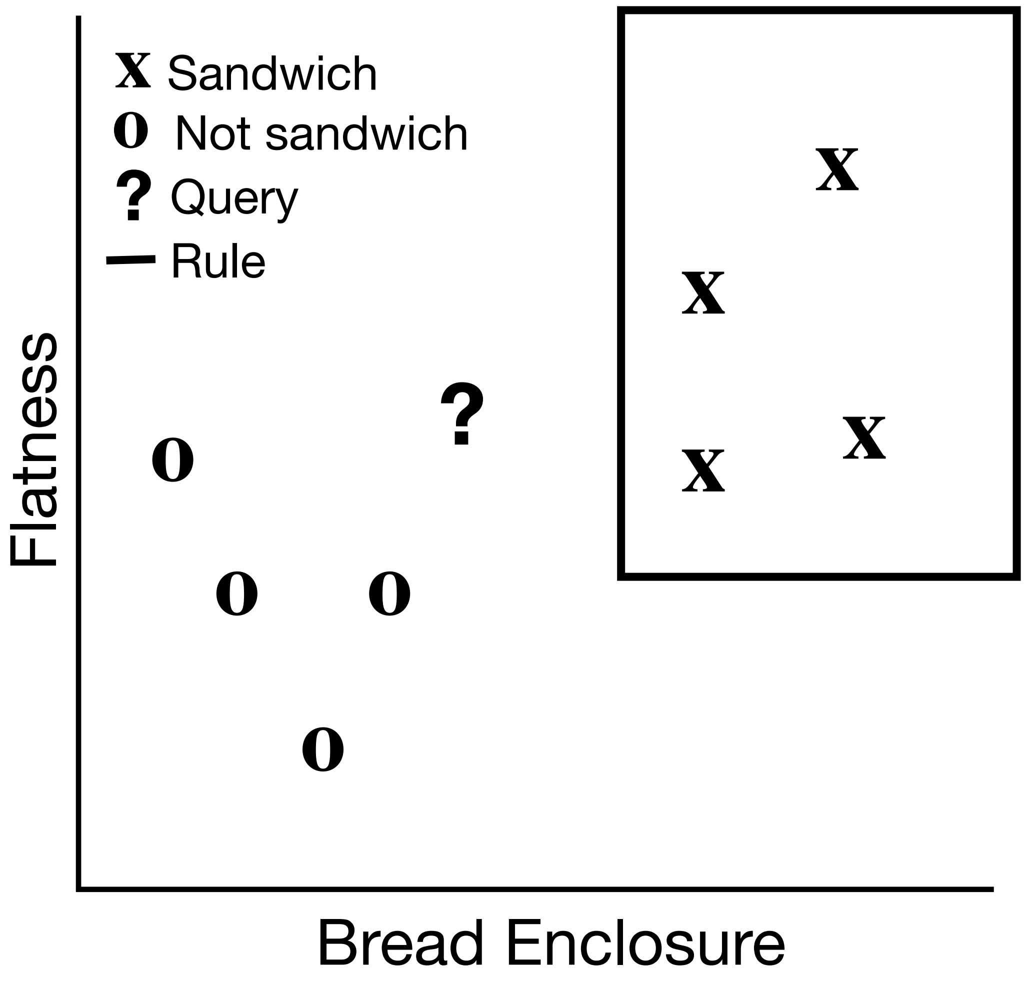


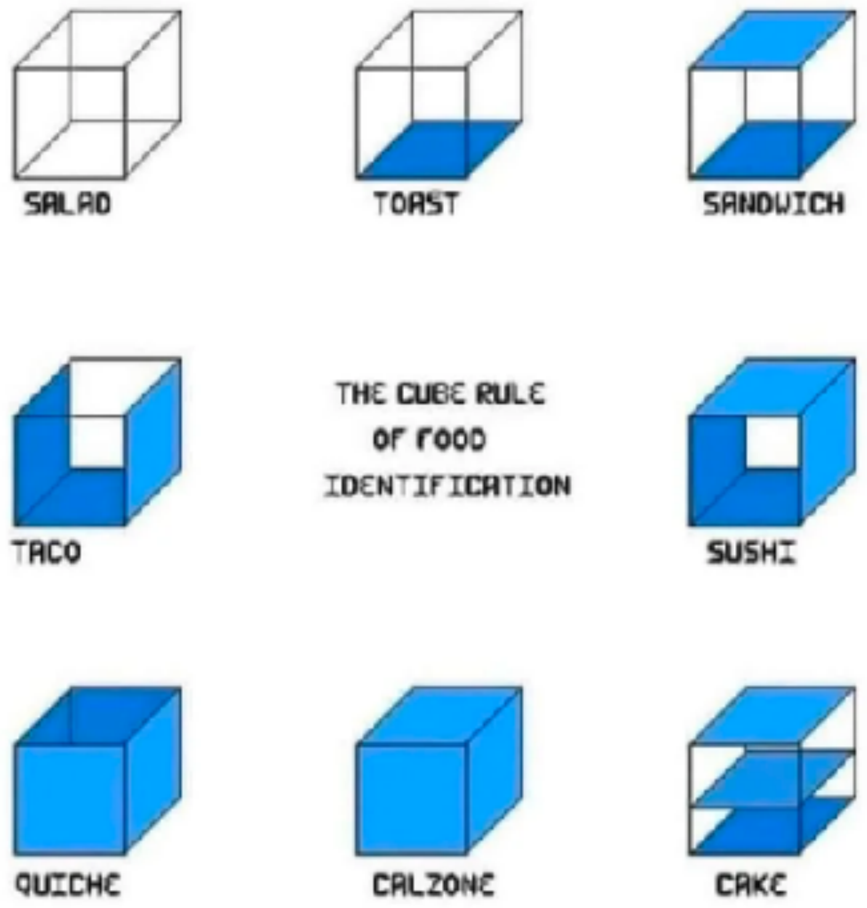
The story so far …

Concept learning as classification

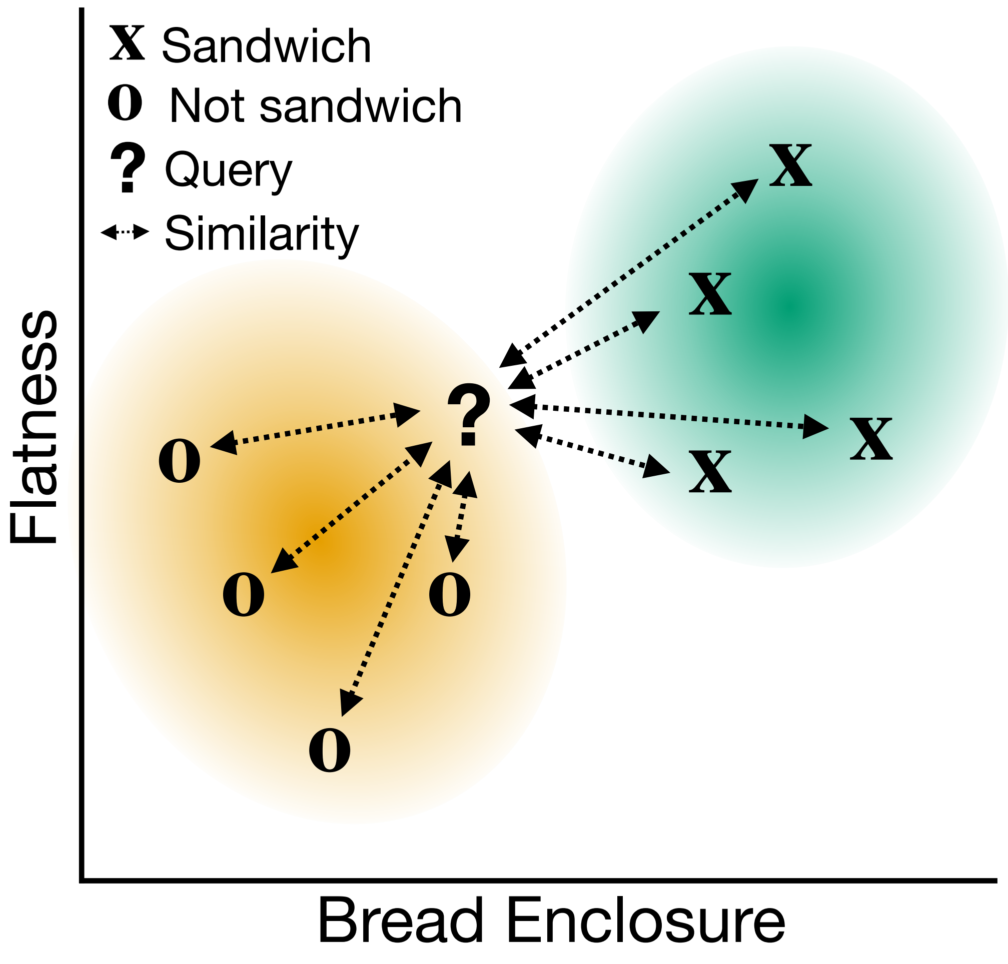


Rule-based





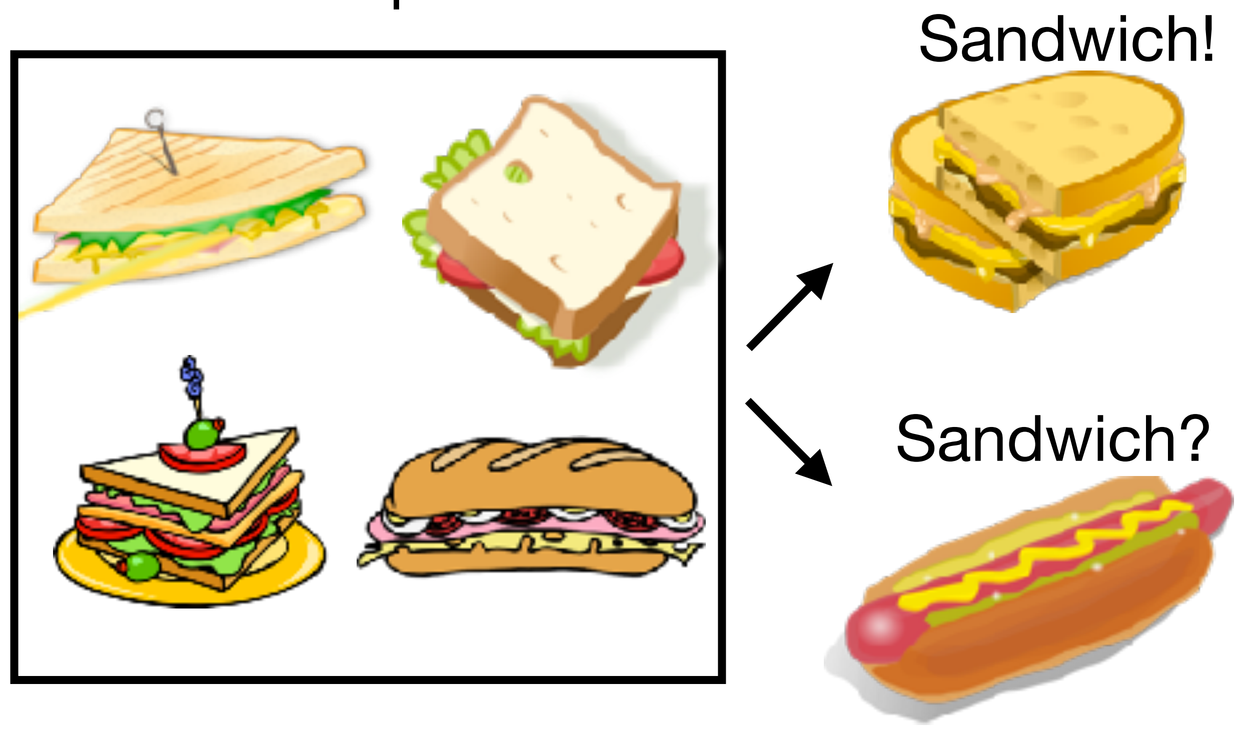
Similarity-based



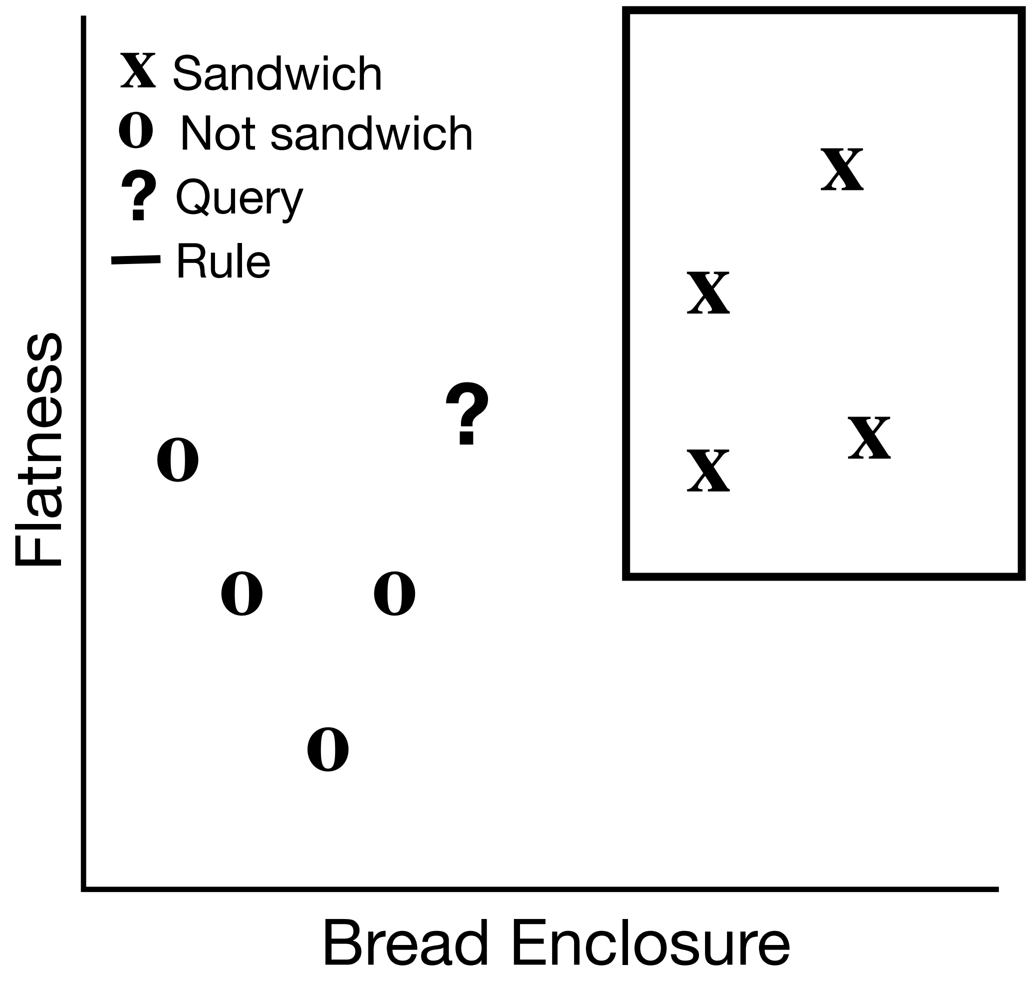
The story so far …

Concept learning as classification

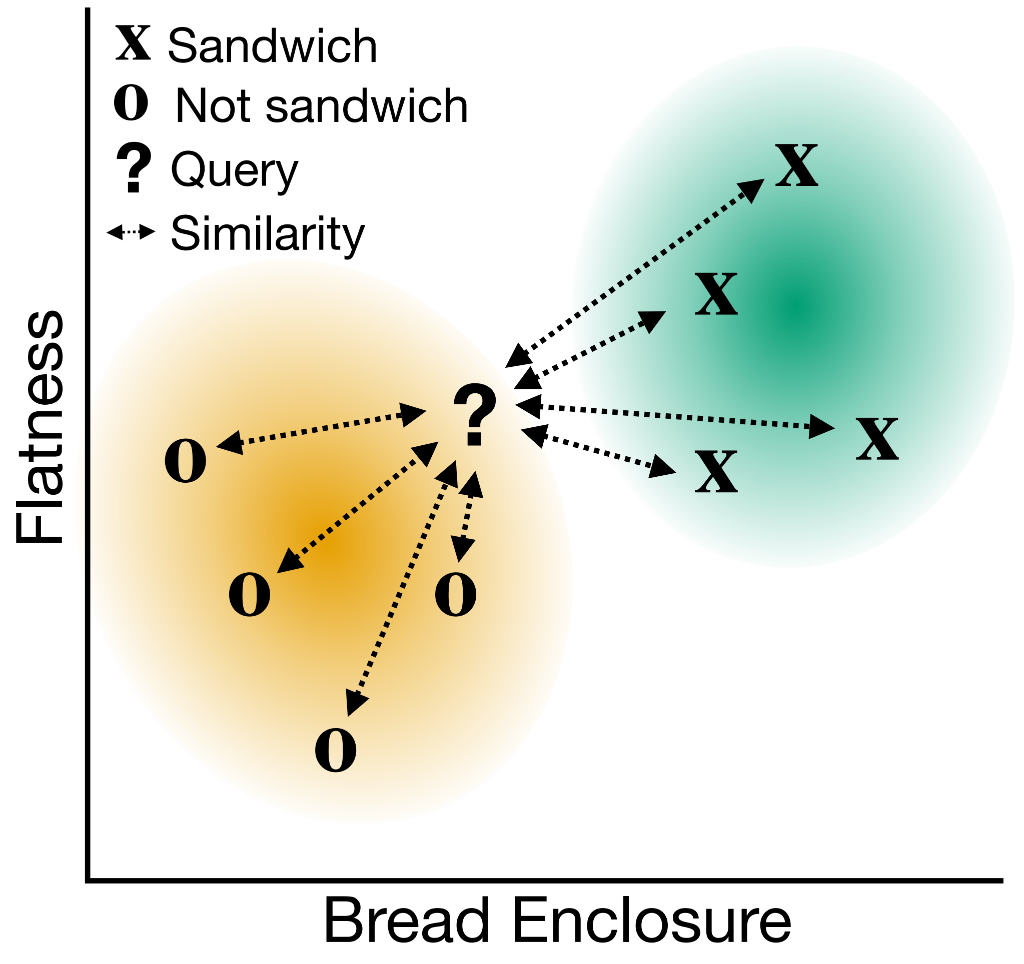
Previous Experiences

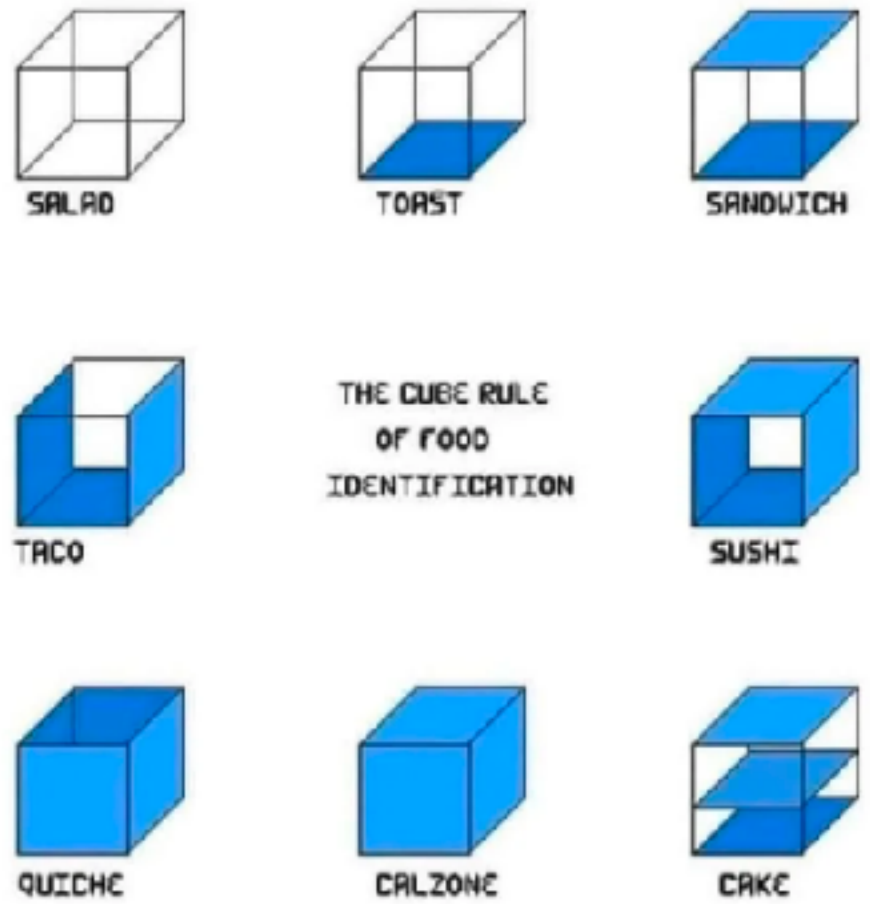


Rule-based

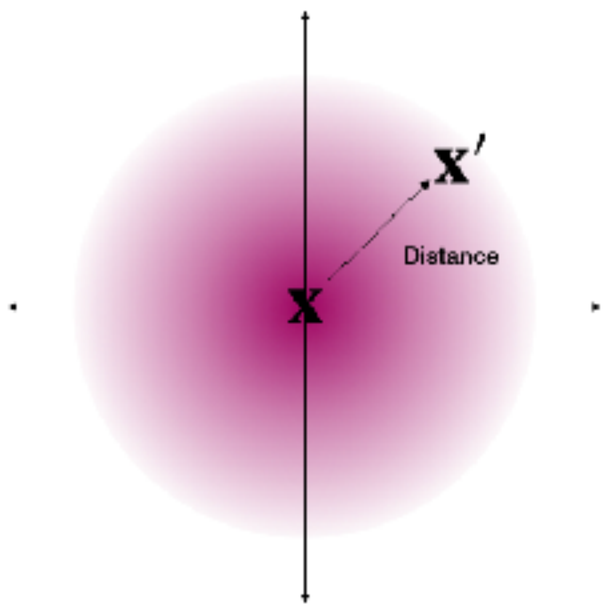


Similarity-based

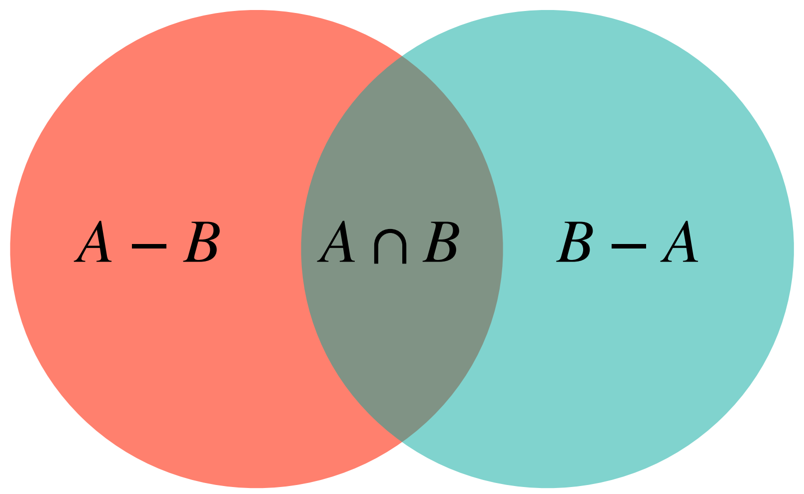




Metric

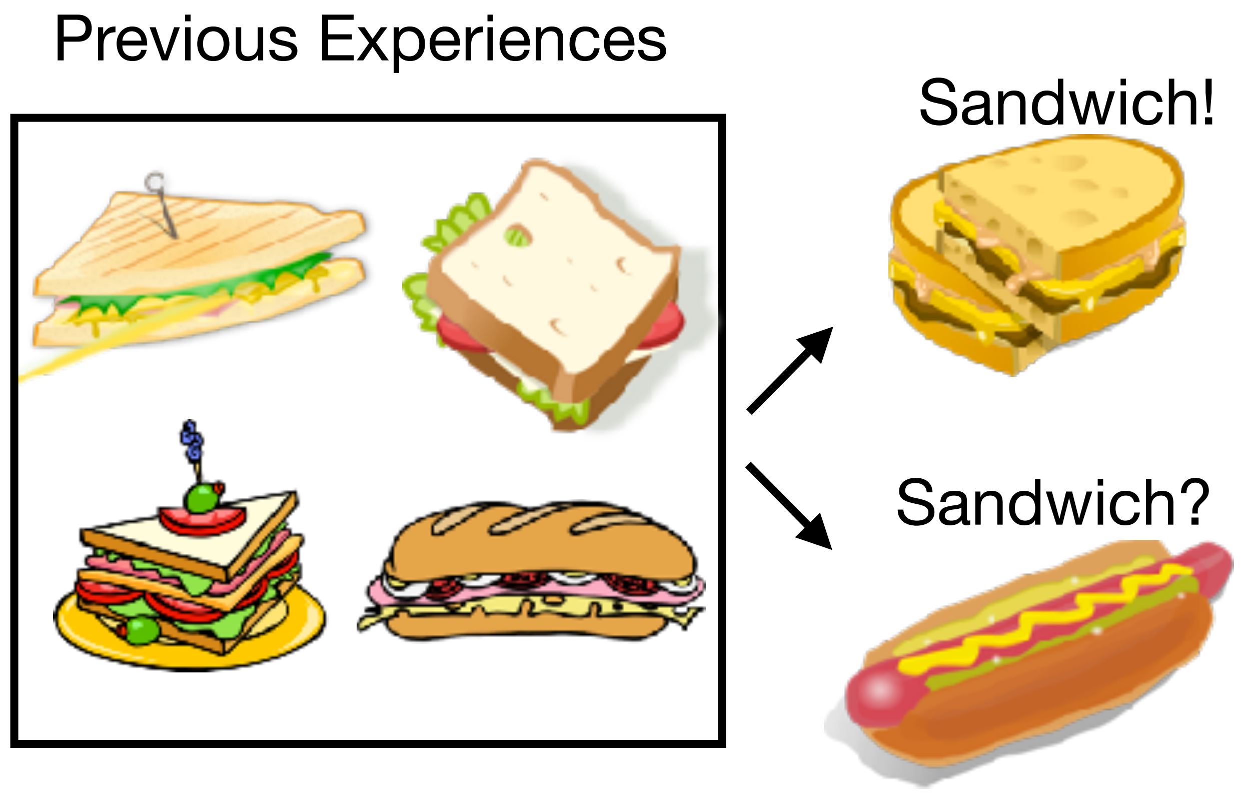


Set

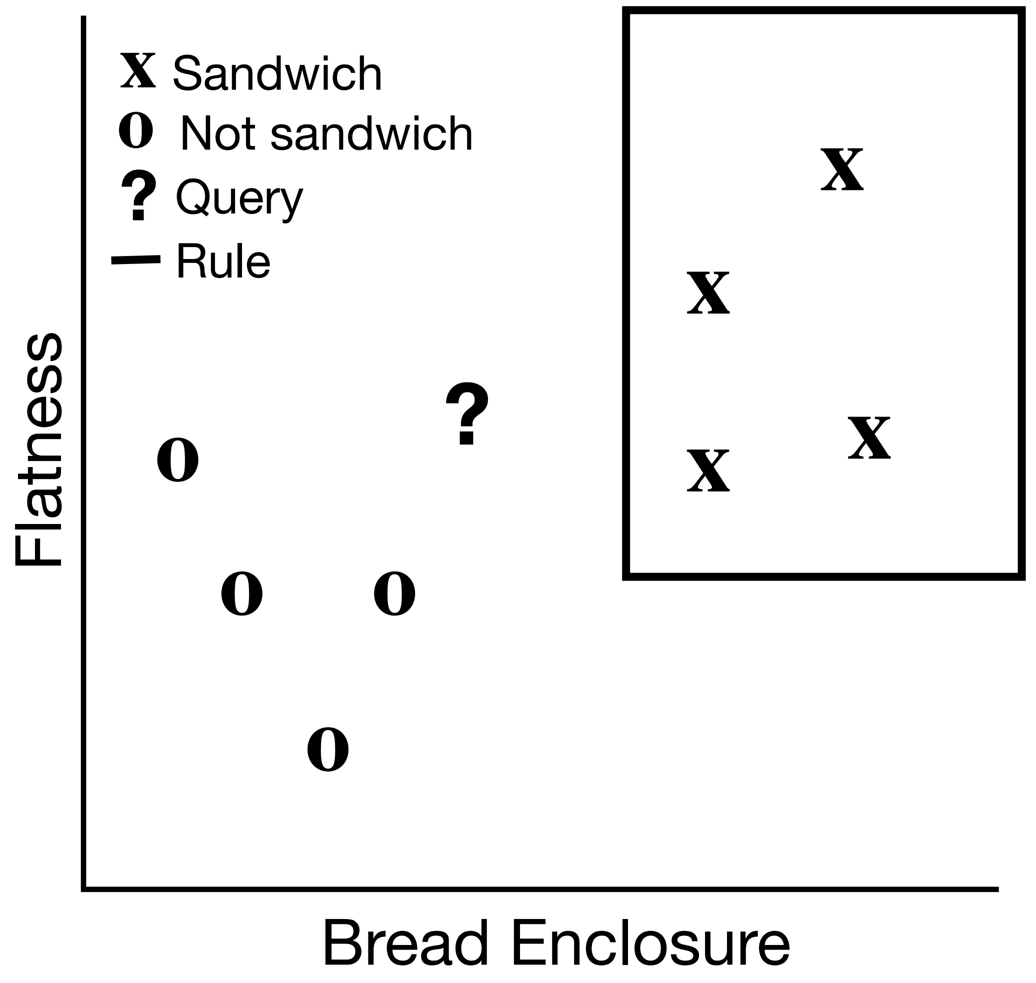


The story so far …

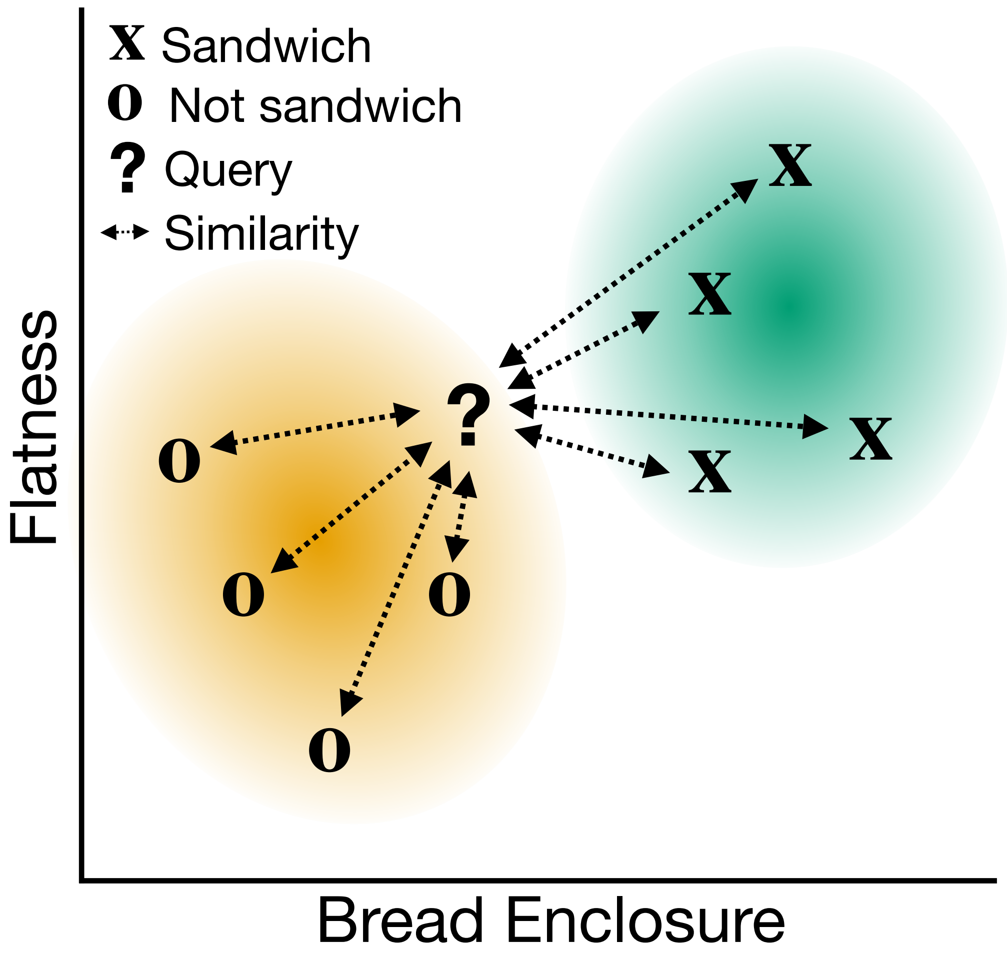
Concept learning as classification



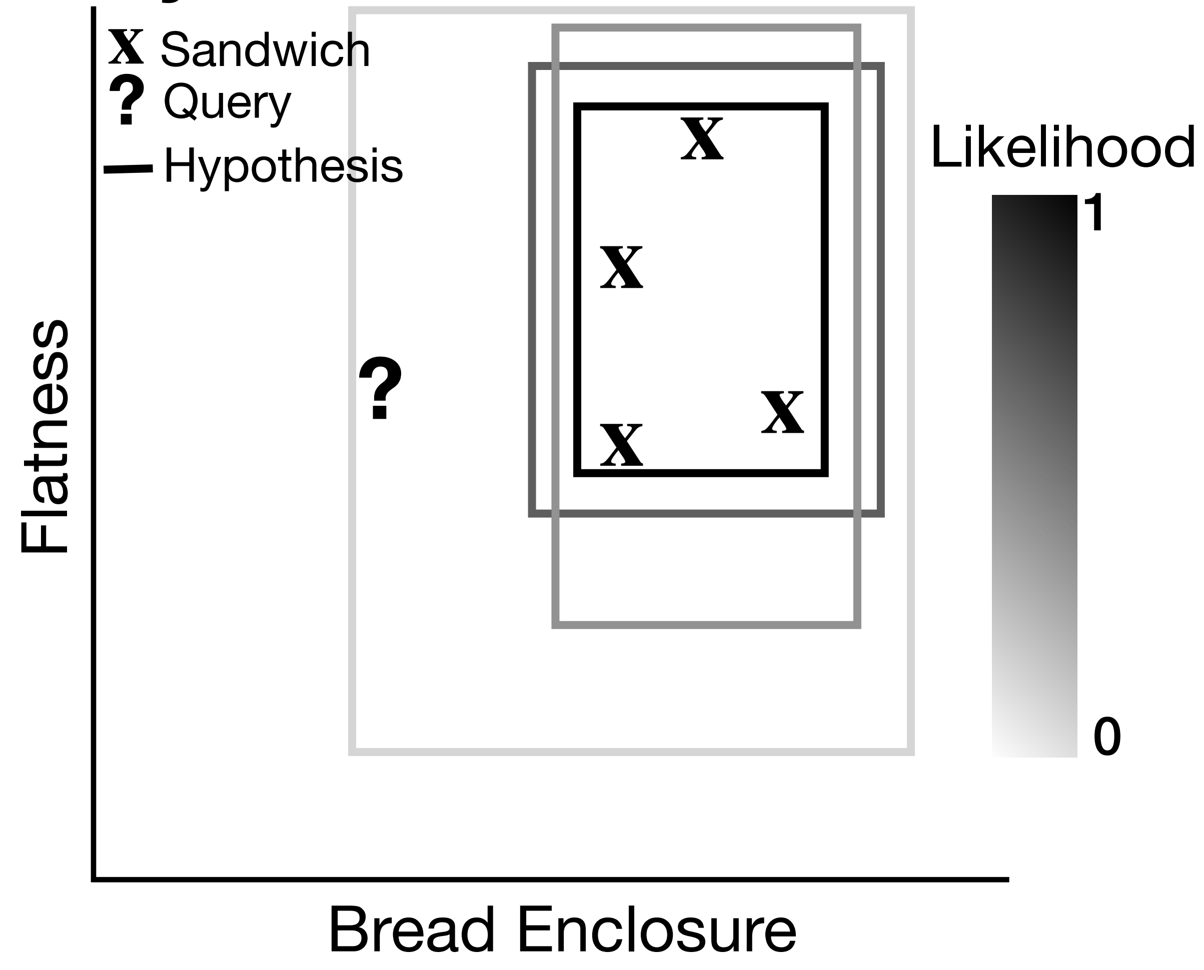
Rule-based

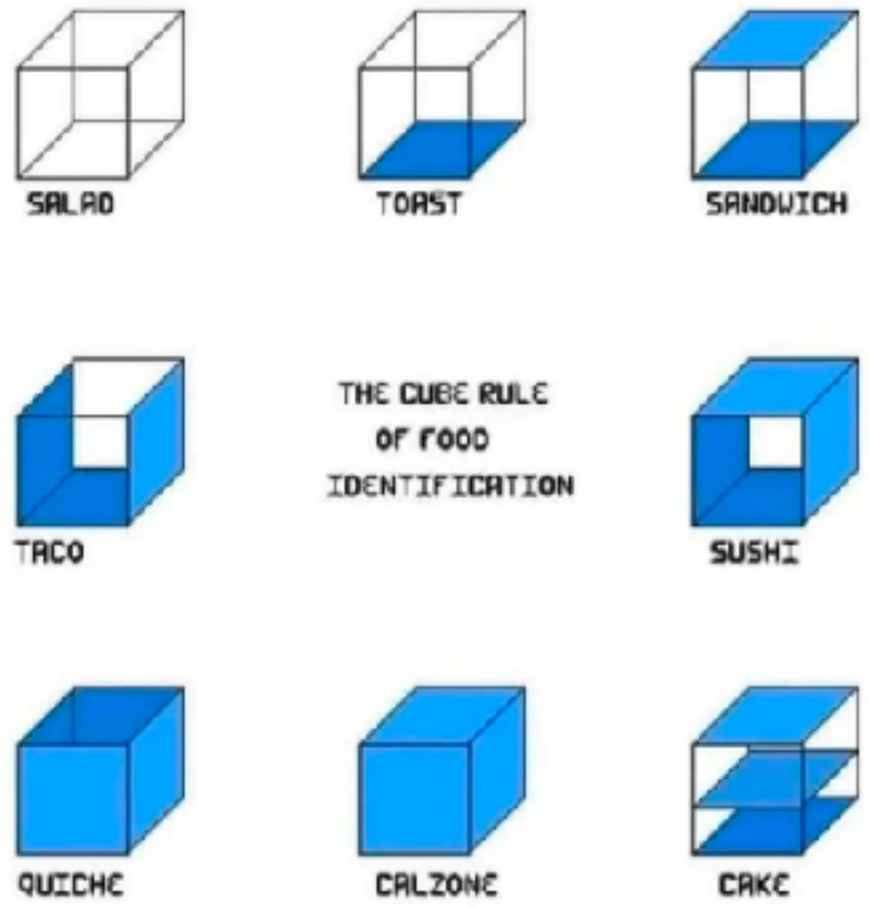


Similarity-based

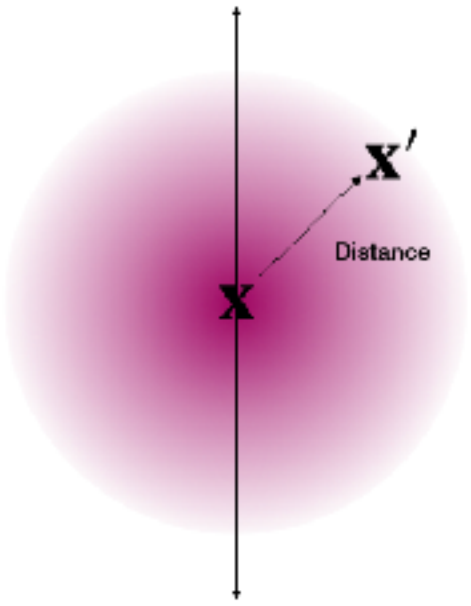


Hybrid

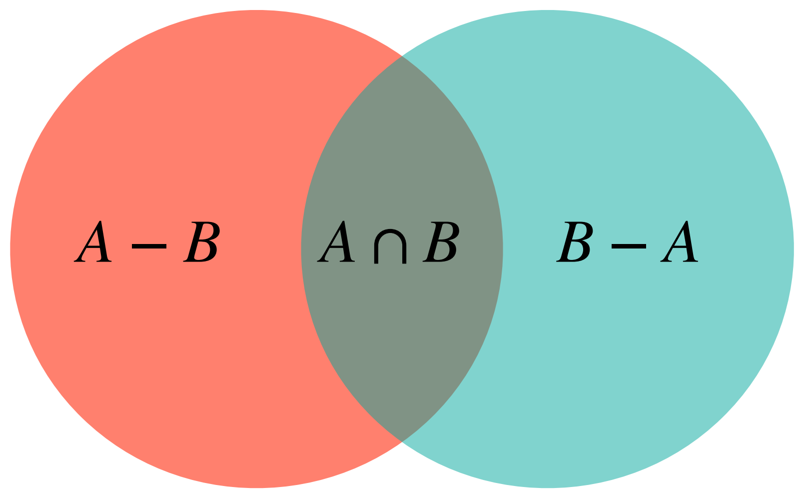




Metric

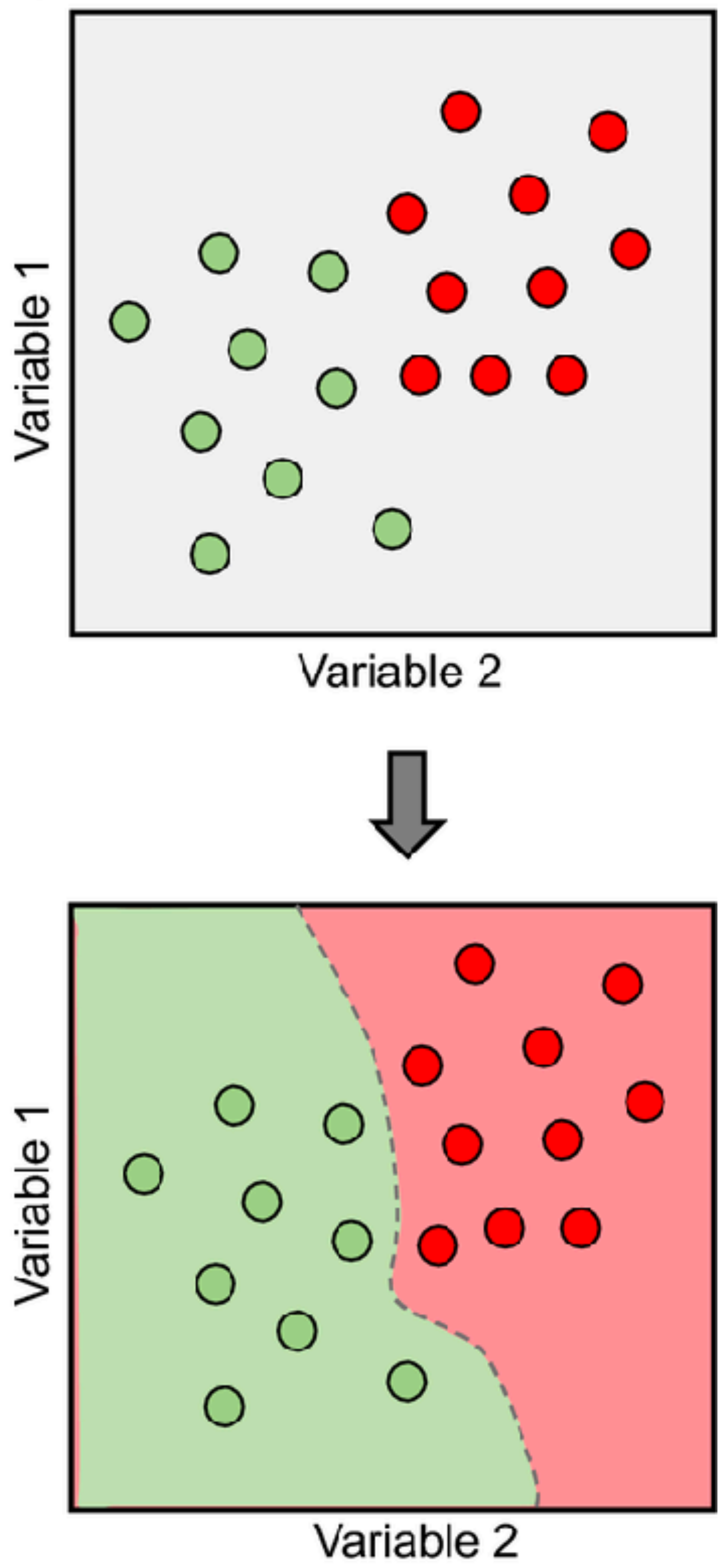


Set

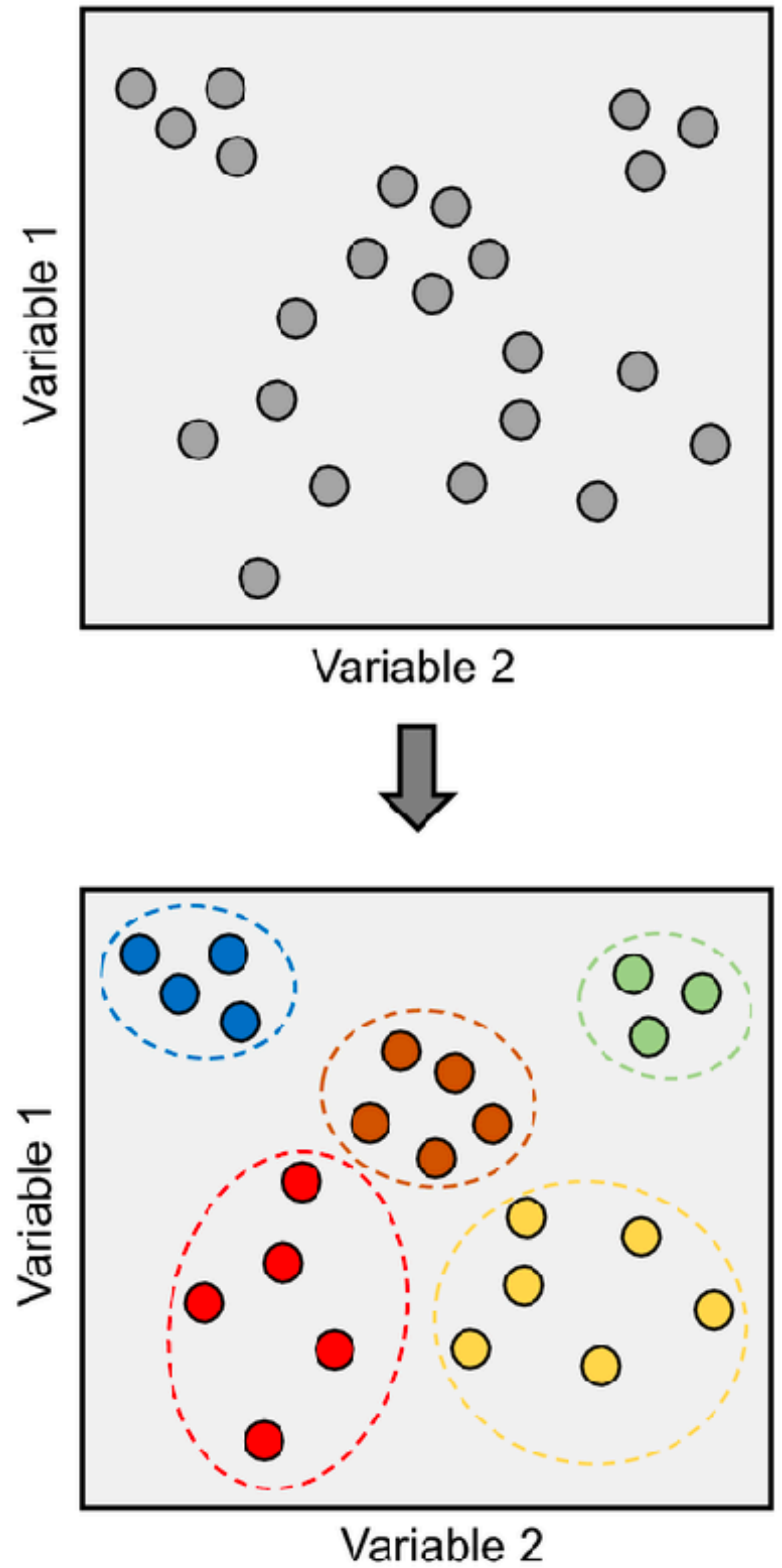


The story so far …

Supervised

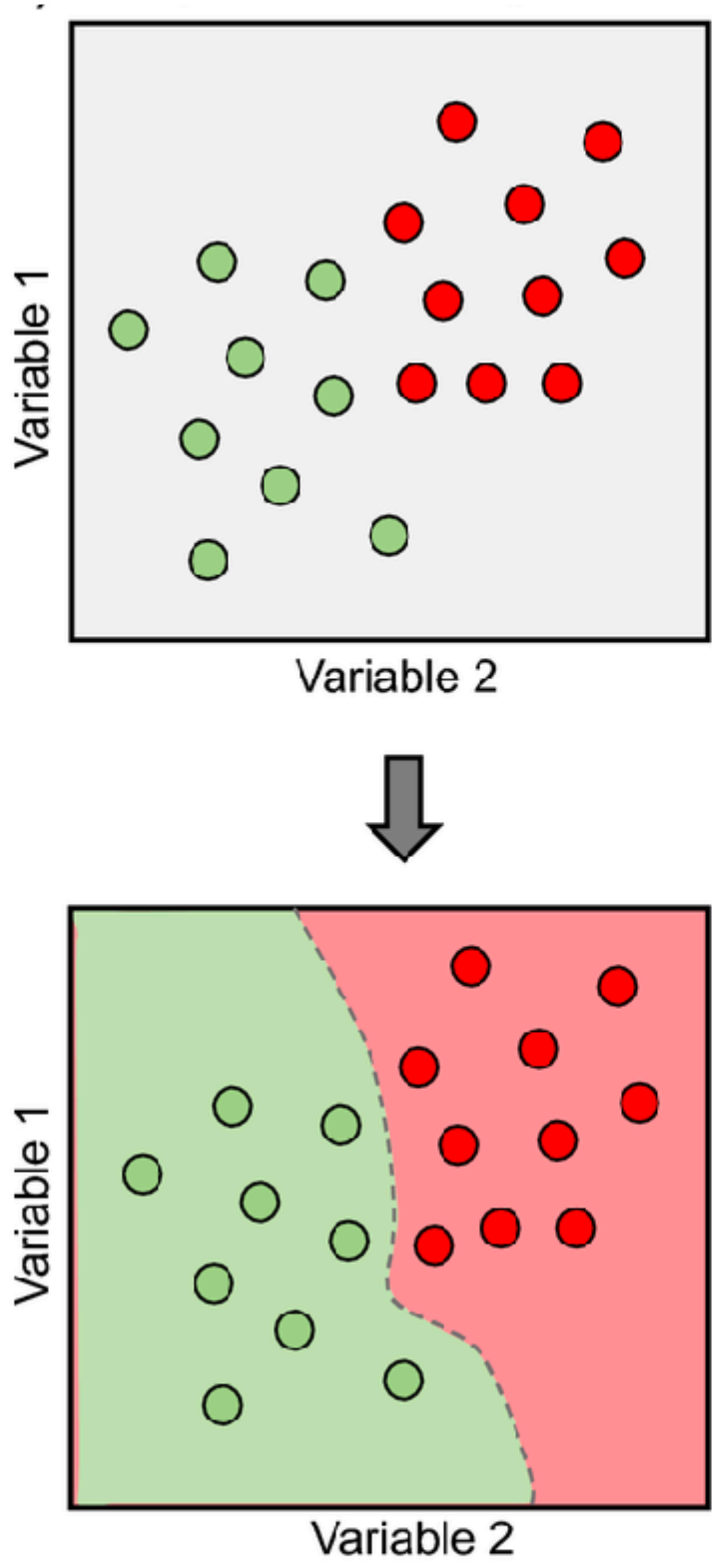


Unsupervised



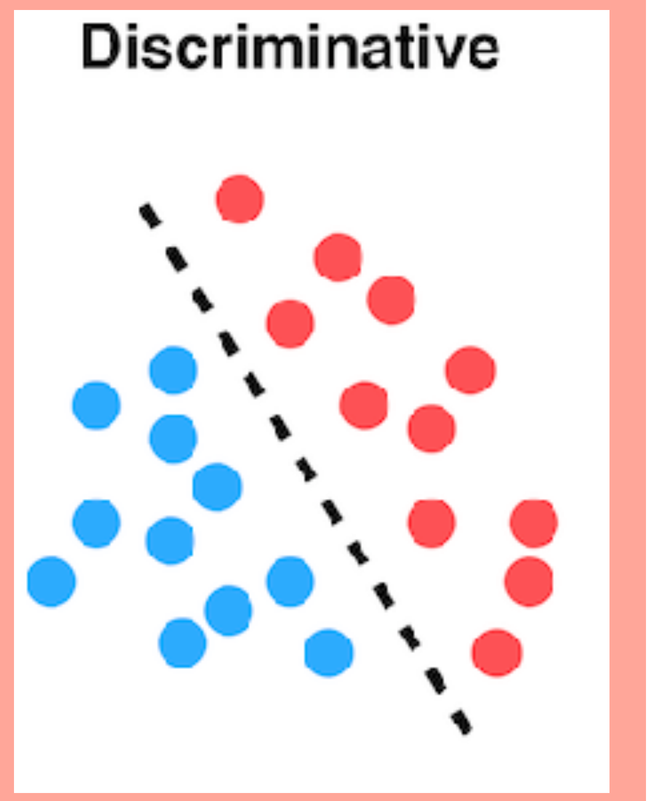
The story so far …

Supervised



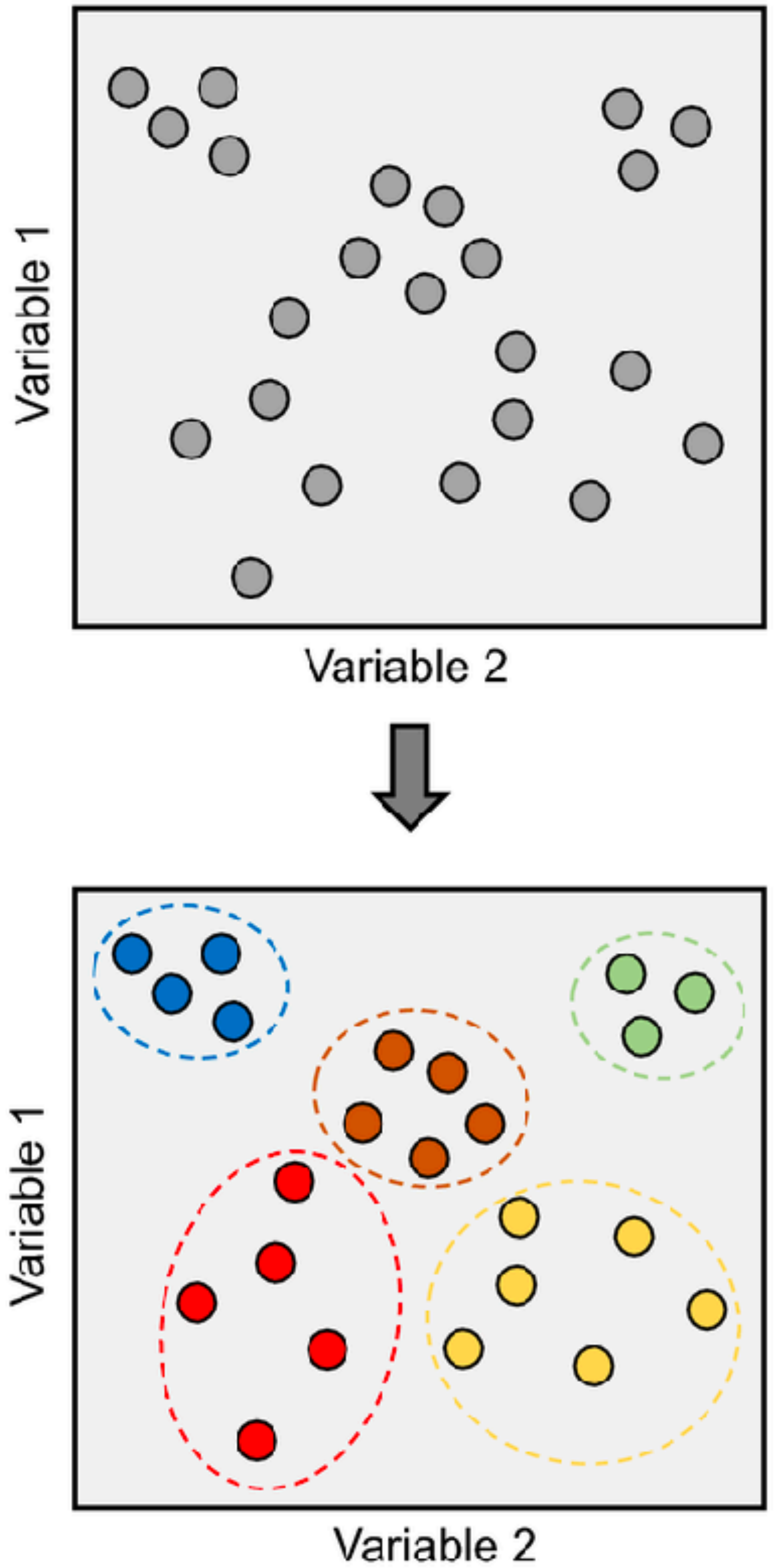
MLPs

Decision trees and random forests



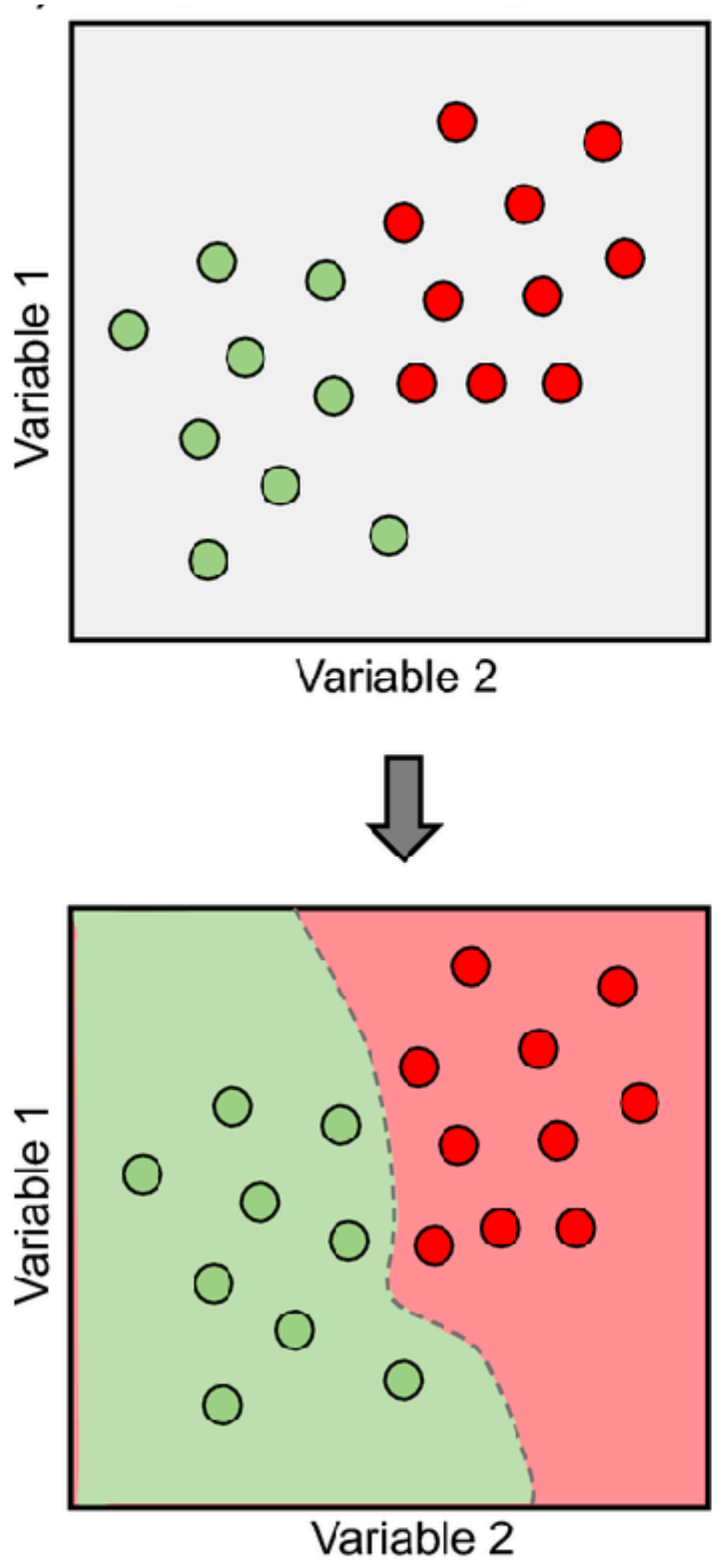
SVMs

Unsupervised

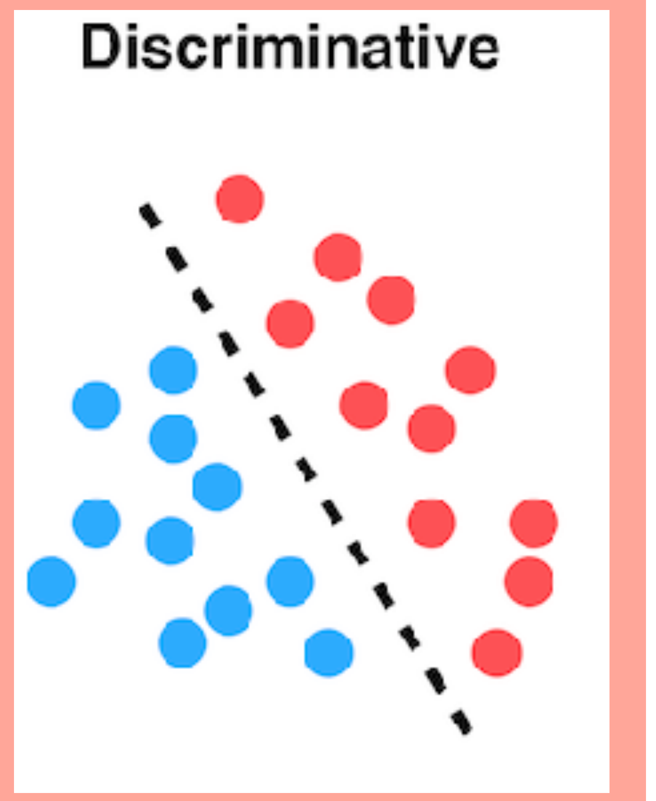


The story so far …

Supervised



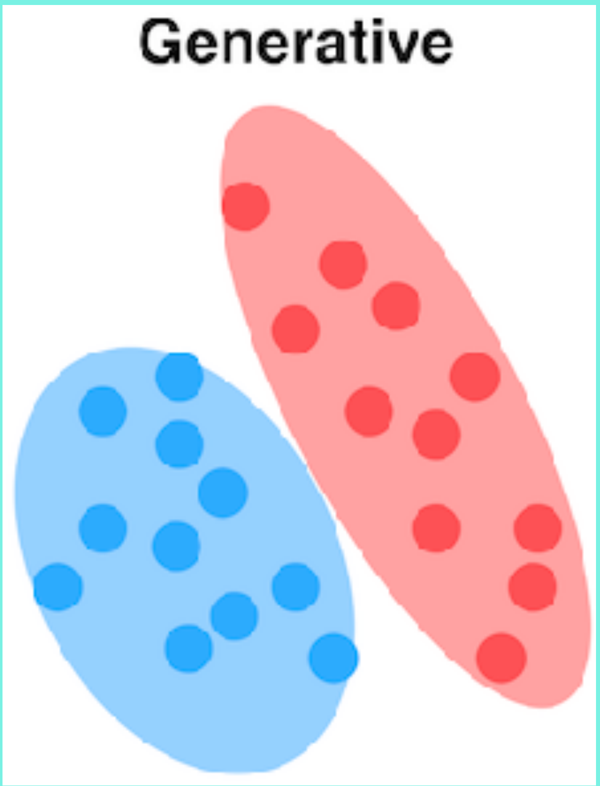
MLPs



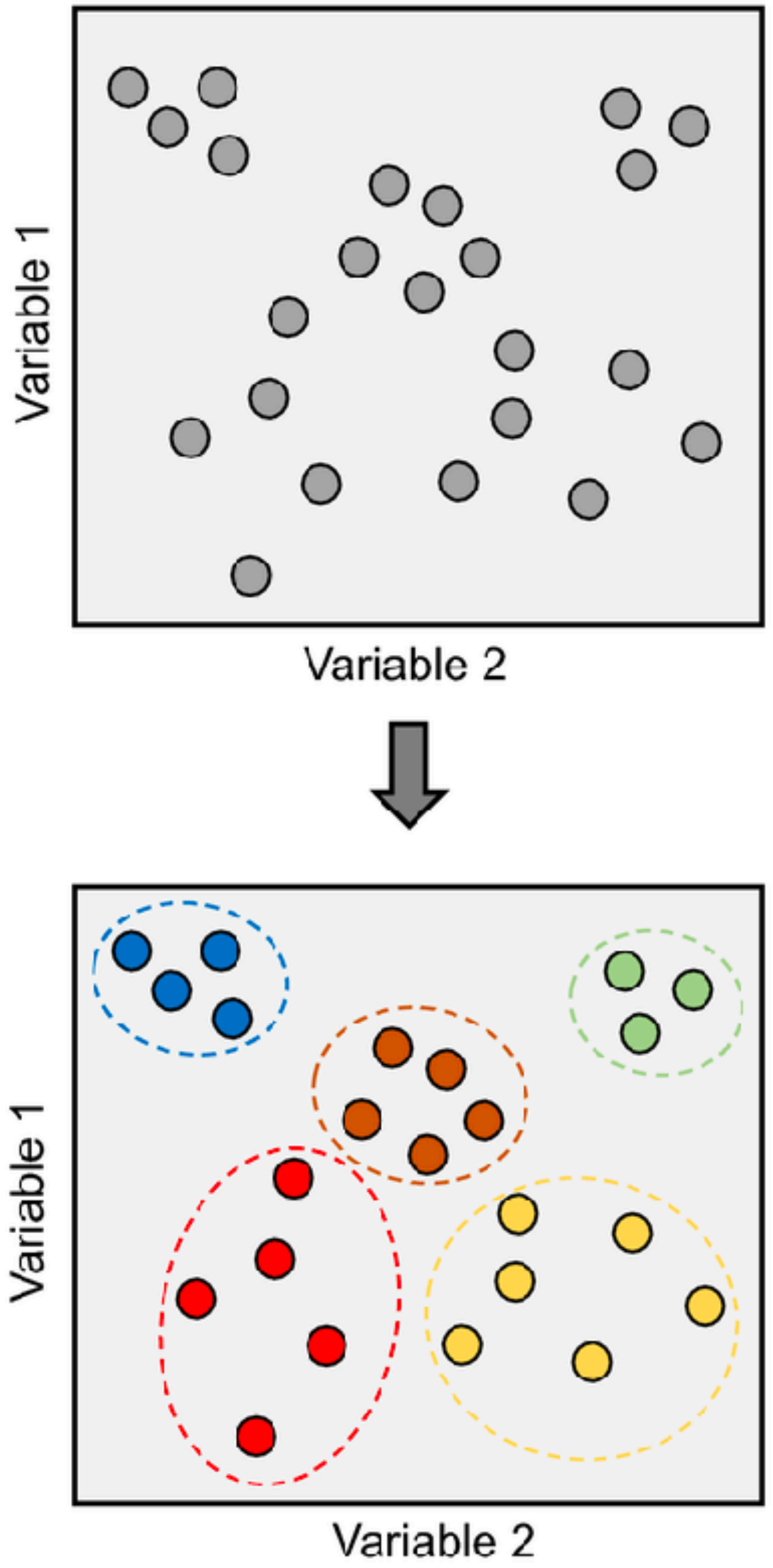
Decision trees and random forests

SVMs

Naïve Bayes

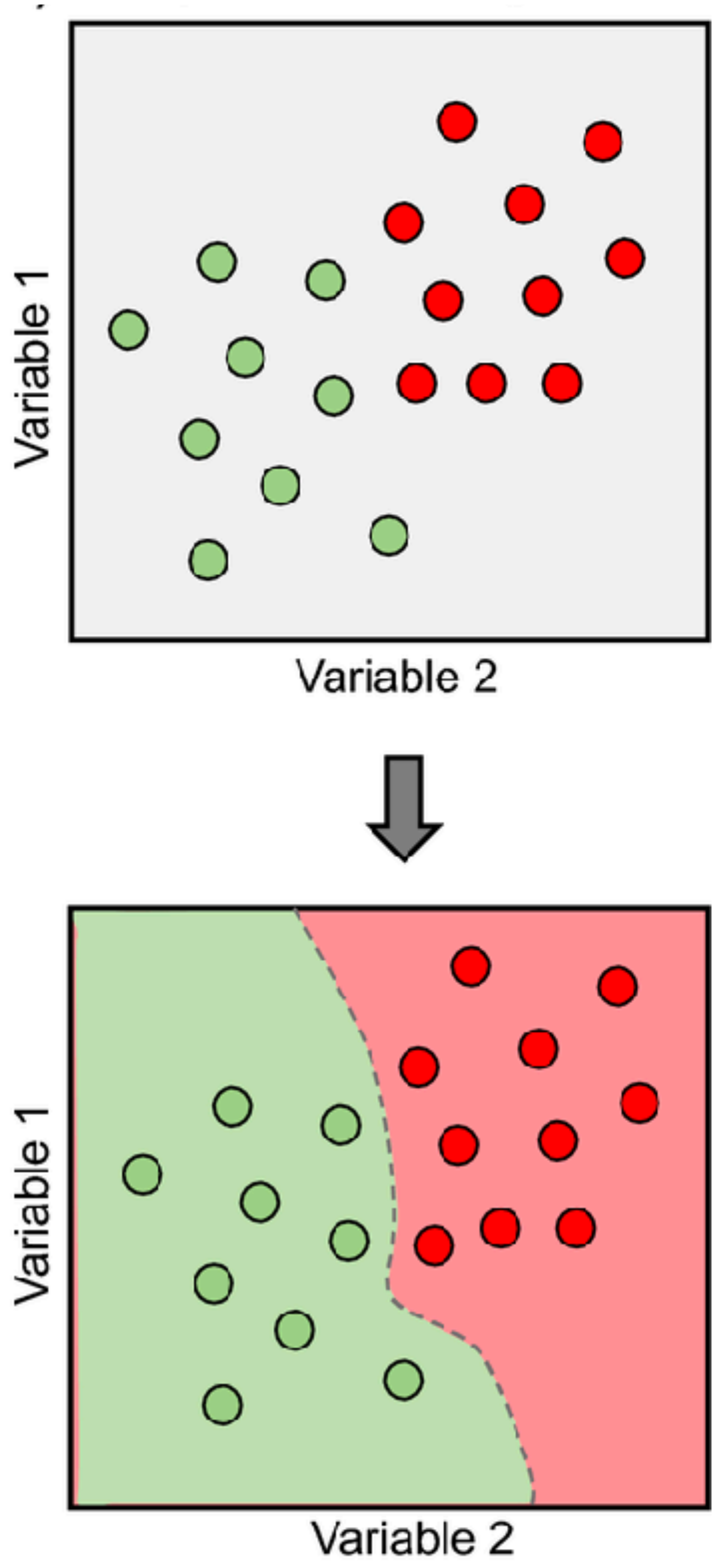


Unsupervised

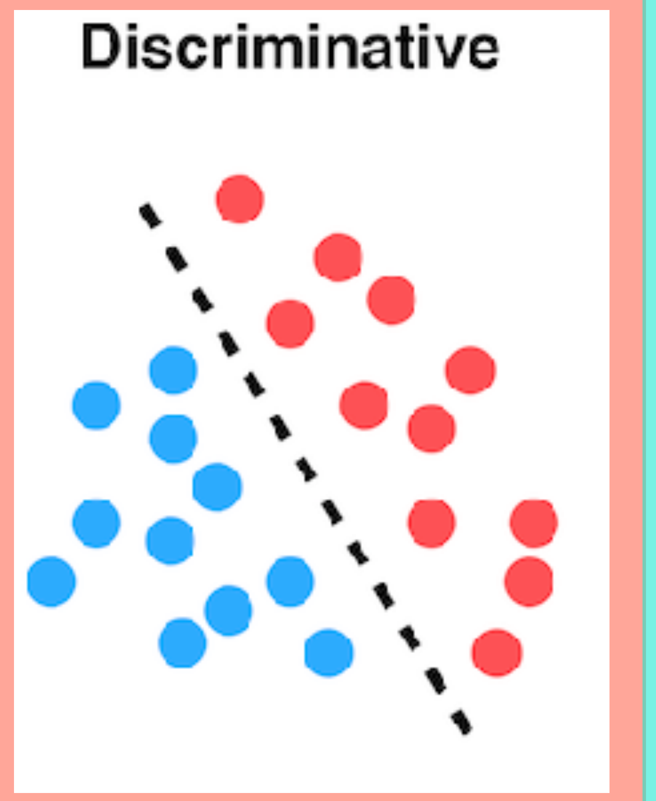


The story so far …

Supervised



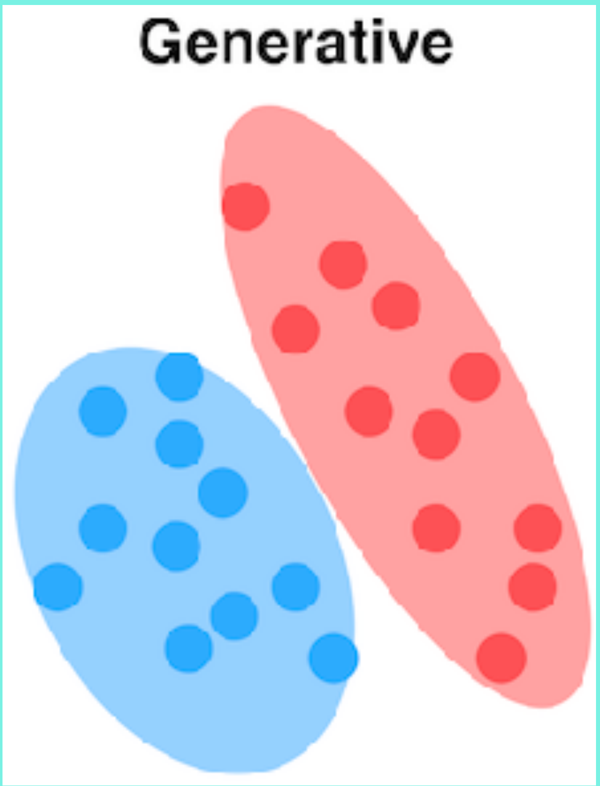
MLPs



Decision trees and random forests

SVMs

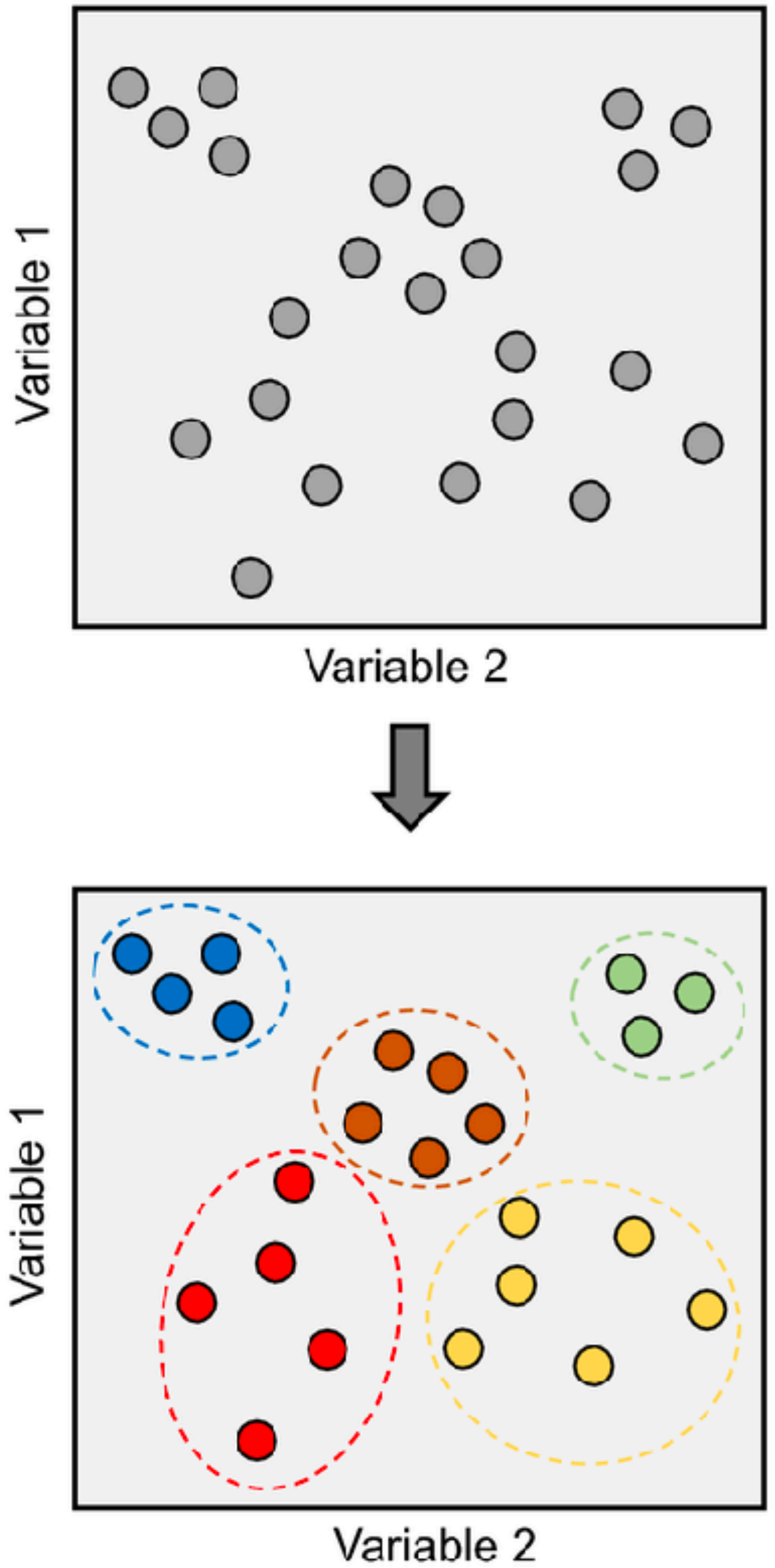
Naïve Bayes



k-Means

GMMs

Unsupervised



From Concepts to Functions

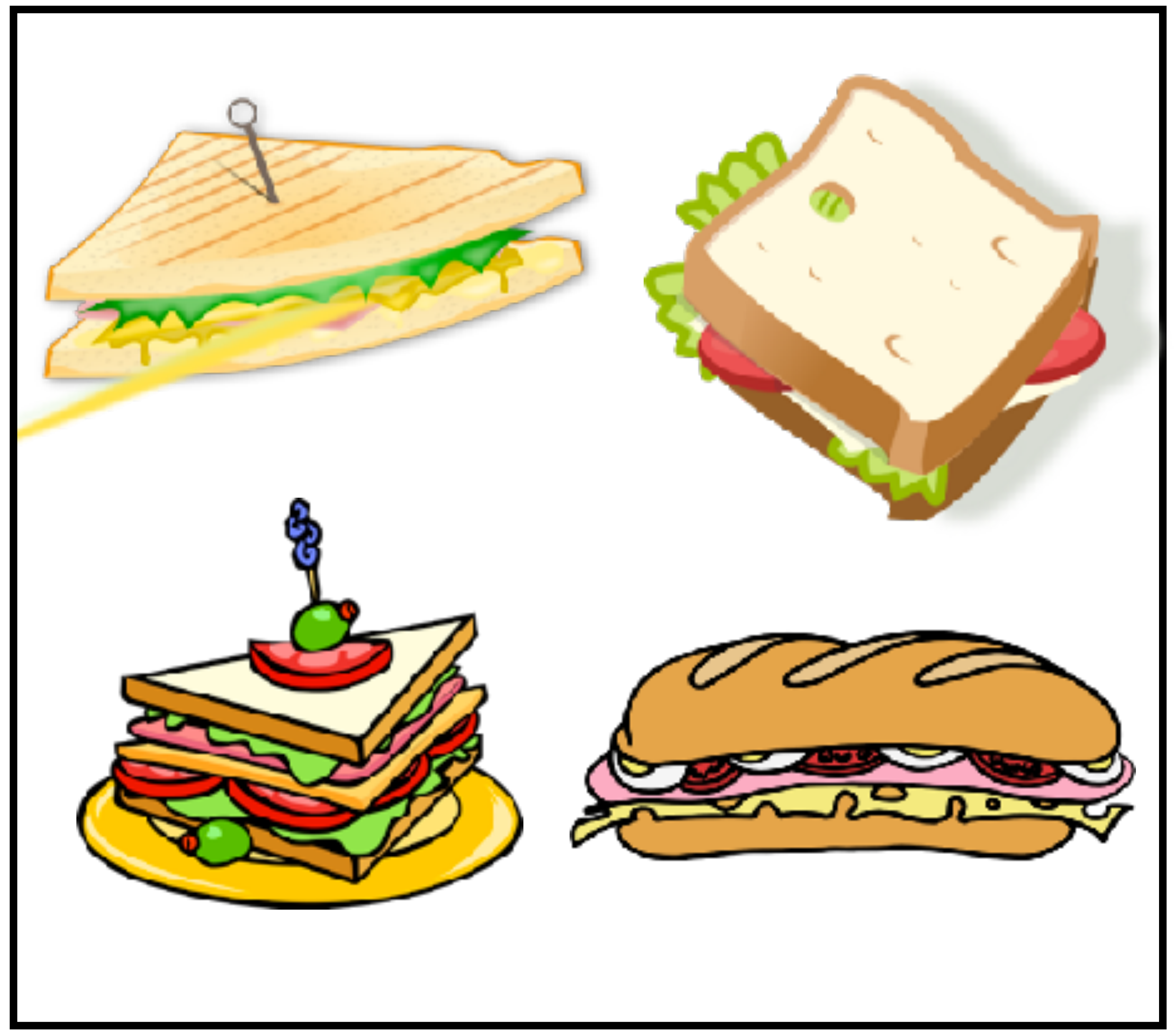
Concept Learning as Classification

Function learning as Regression

From Concepts to Functions

Concept Learning as Classification

Previous Experiences

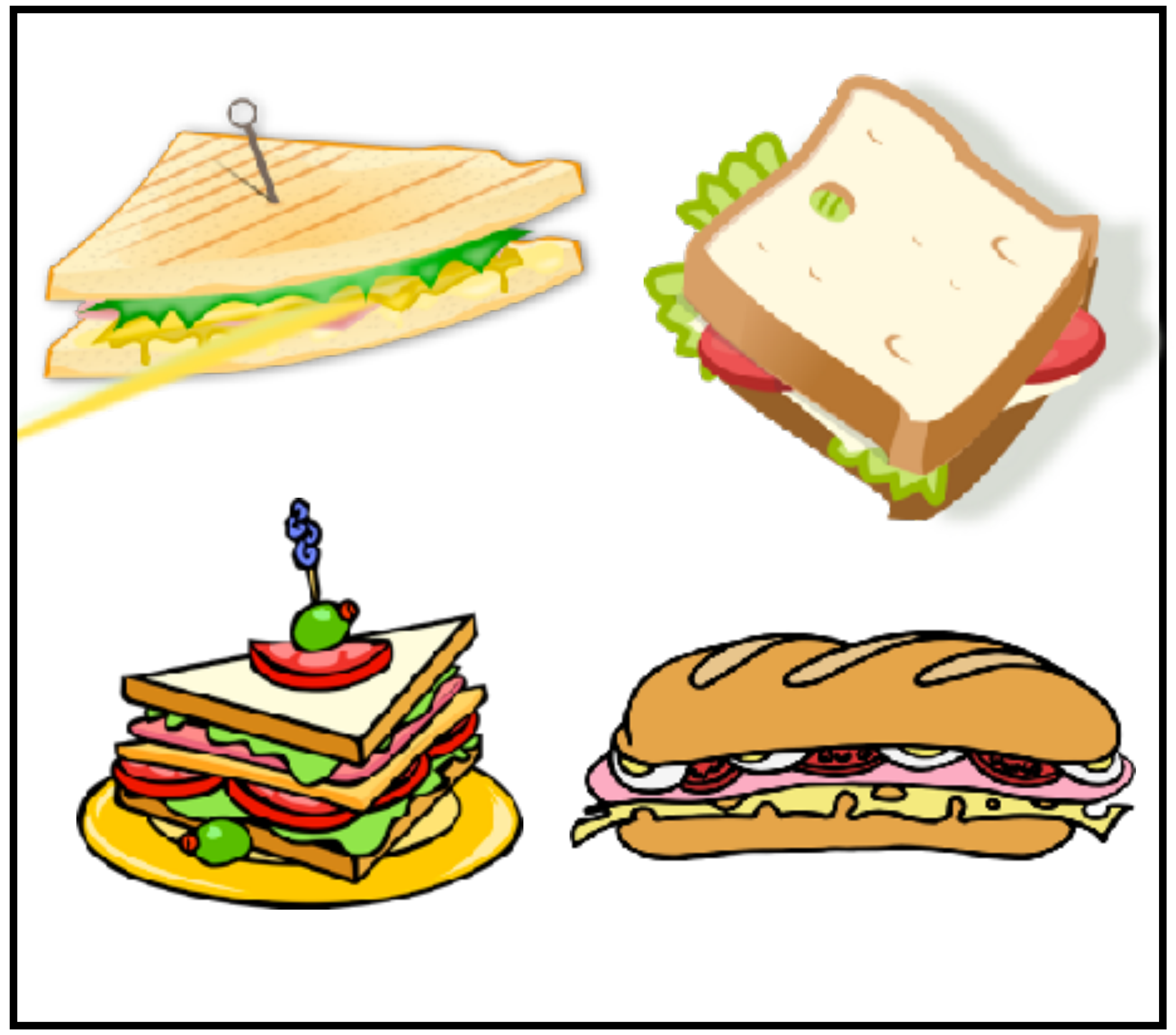


Function learning as Regression

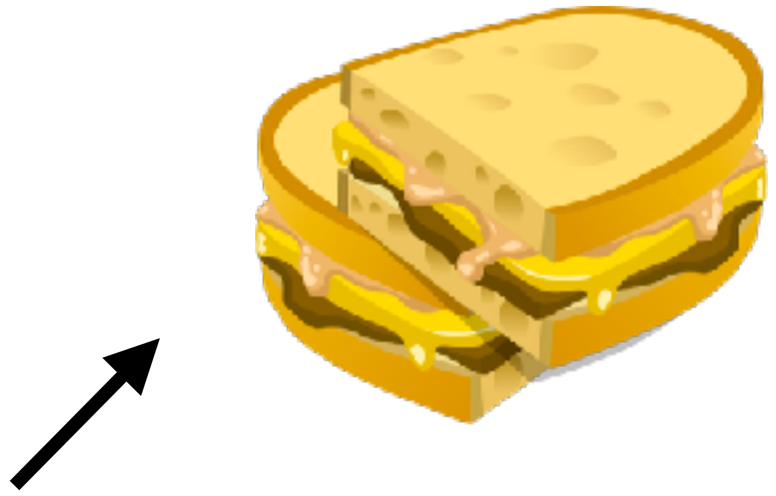
From Concepts to Functions

Concept Learning as Classification

Previous Experiences



Sandwich!

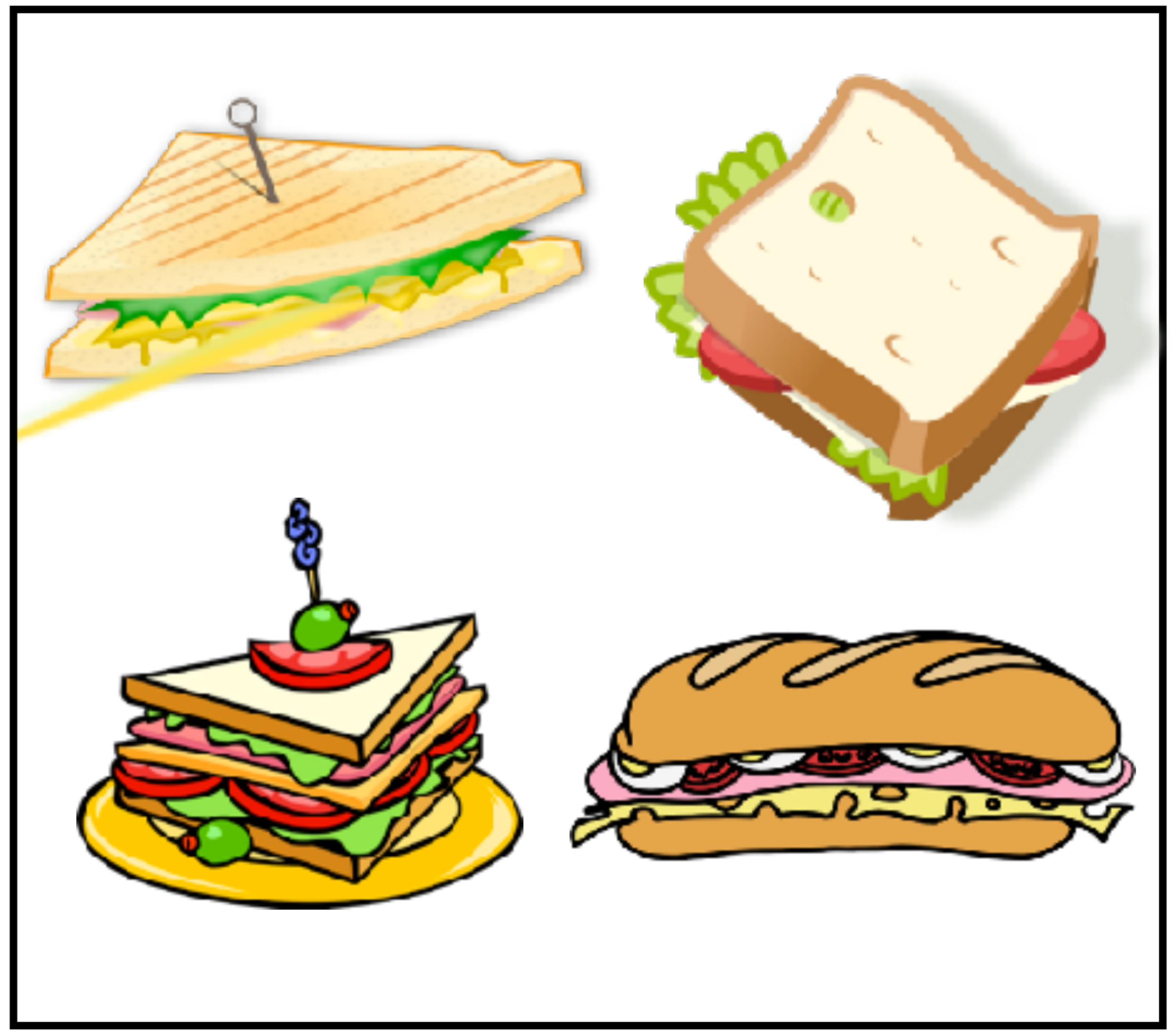


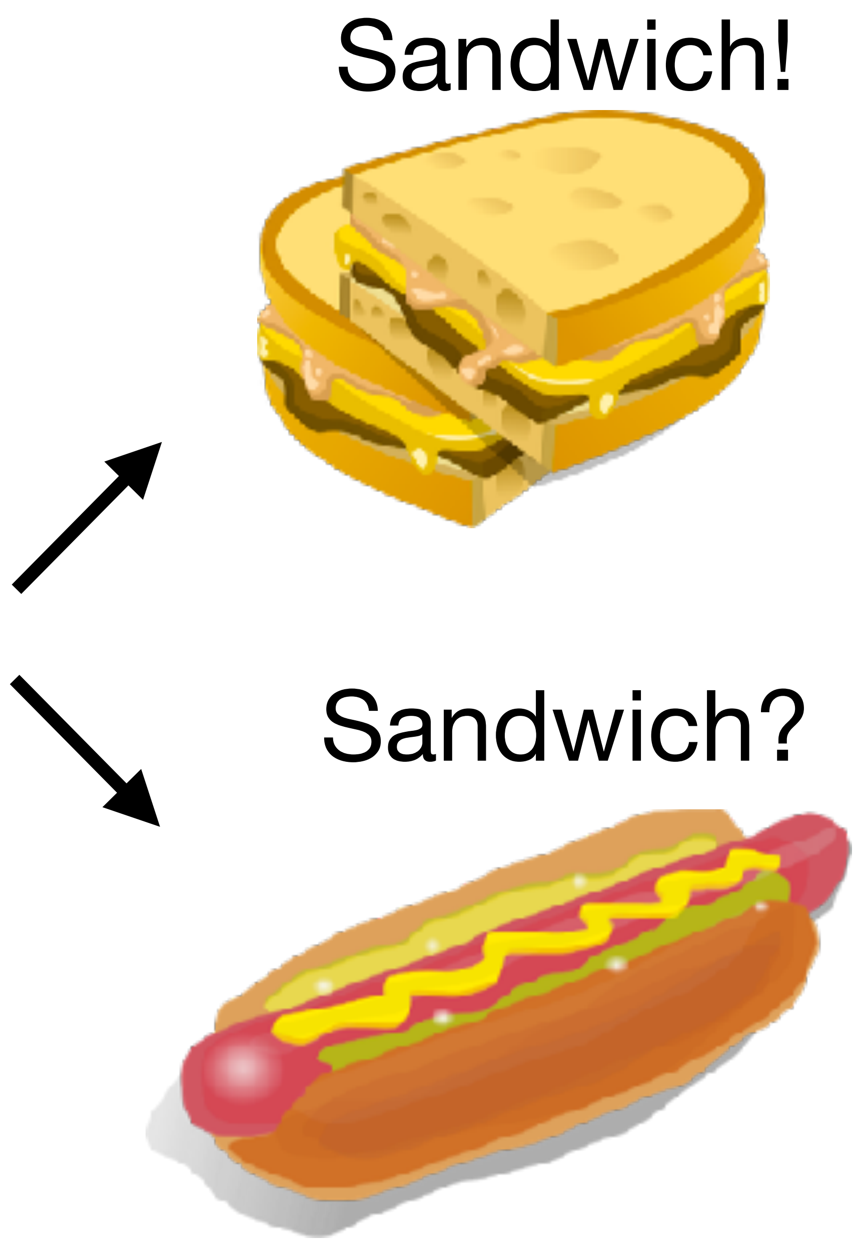
Function learning as Regression

From Concepts to Functions

Concept Learning as Classification

Previous Experiences



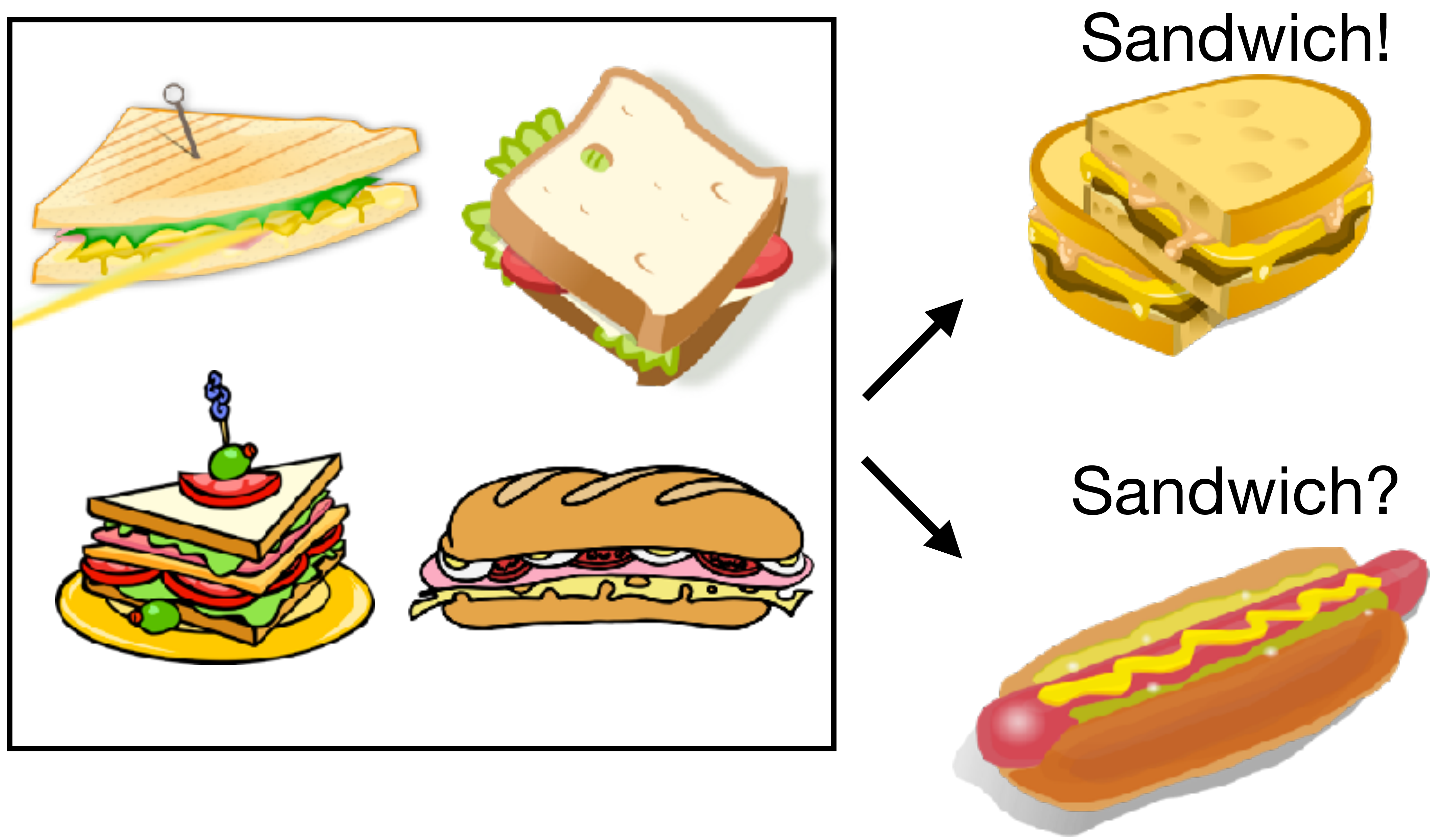


Function learning as Regression

From Concepts to Functions

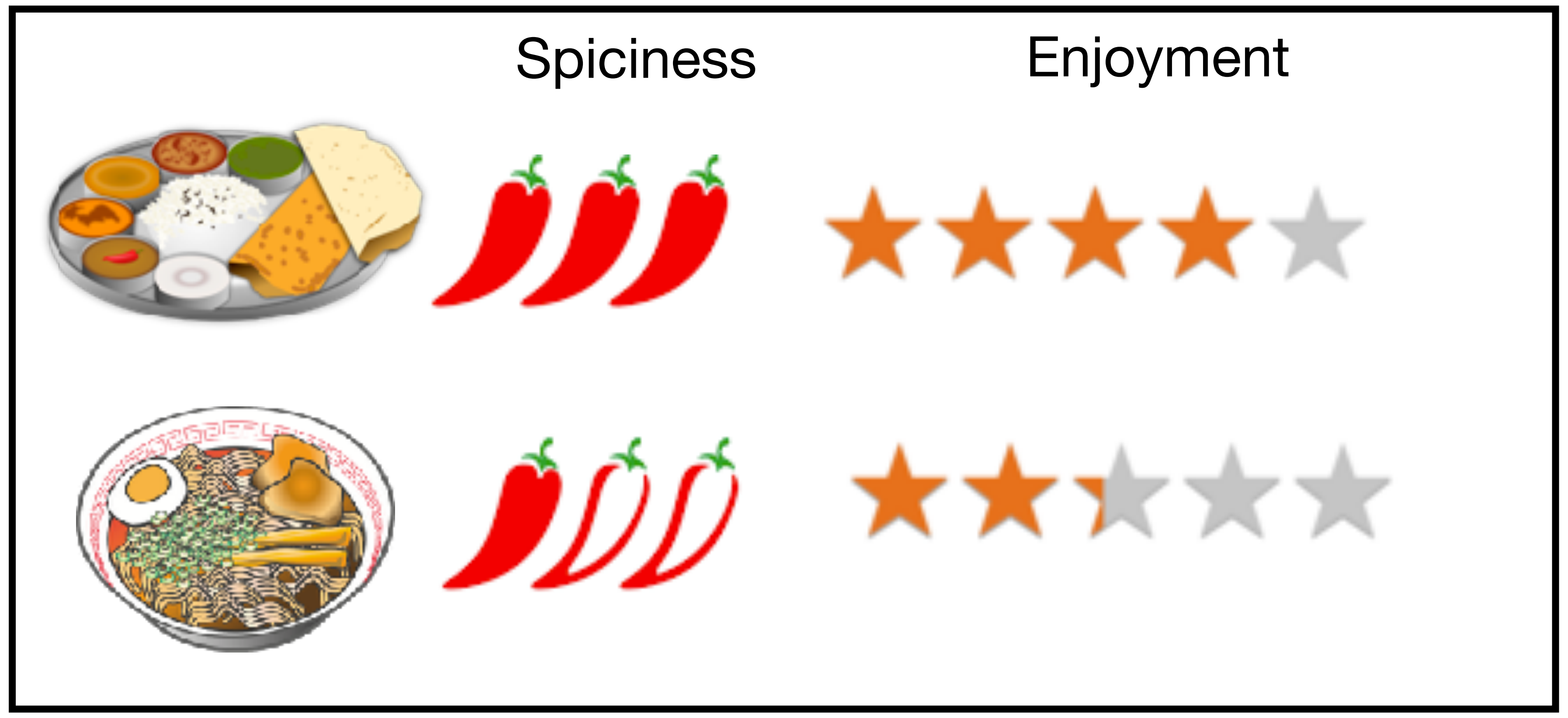
Concept Learning as Classification

Previous Experiences



Function learning as Regression

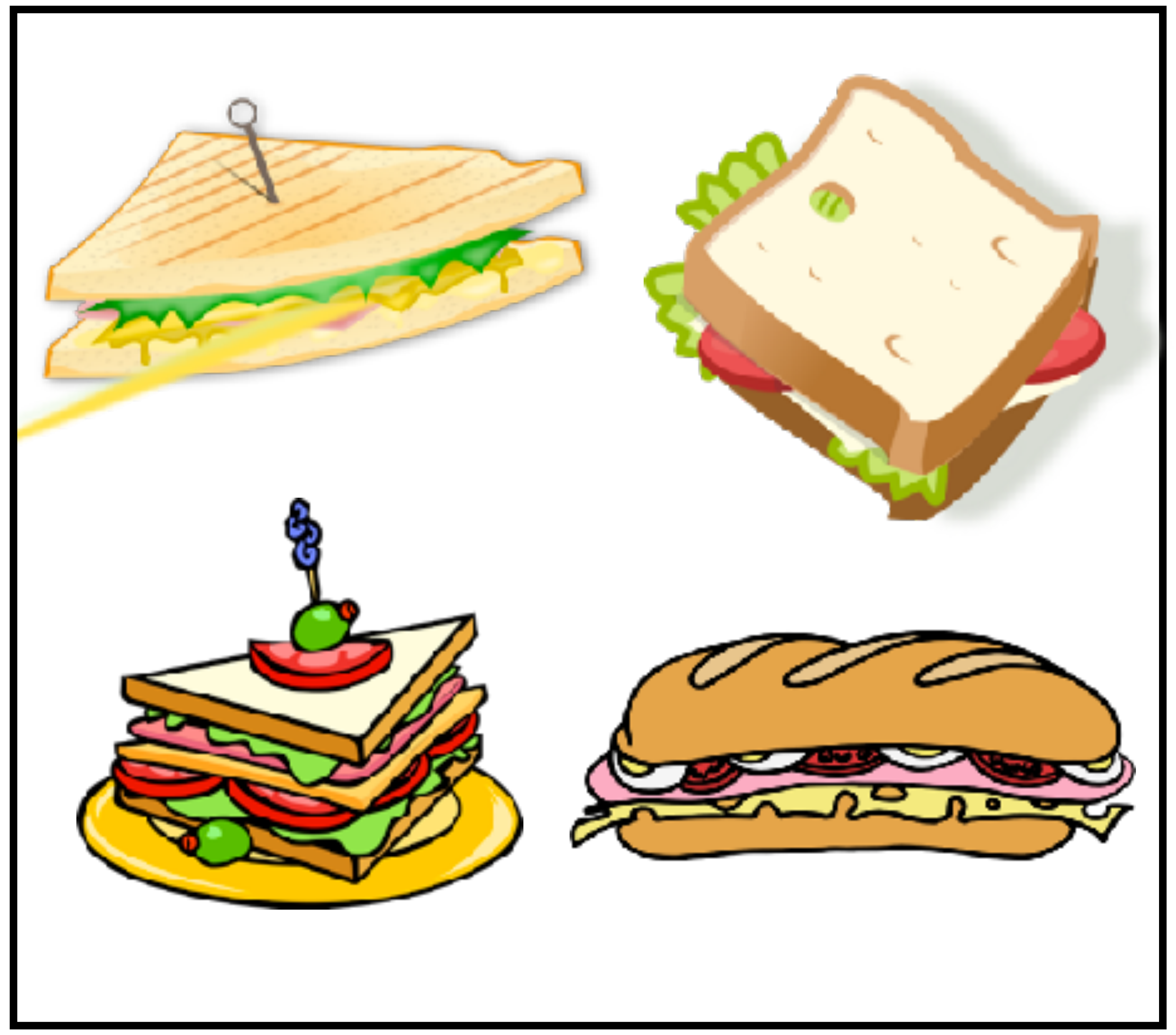
Previous Experiences



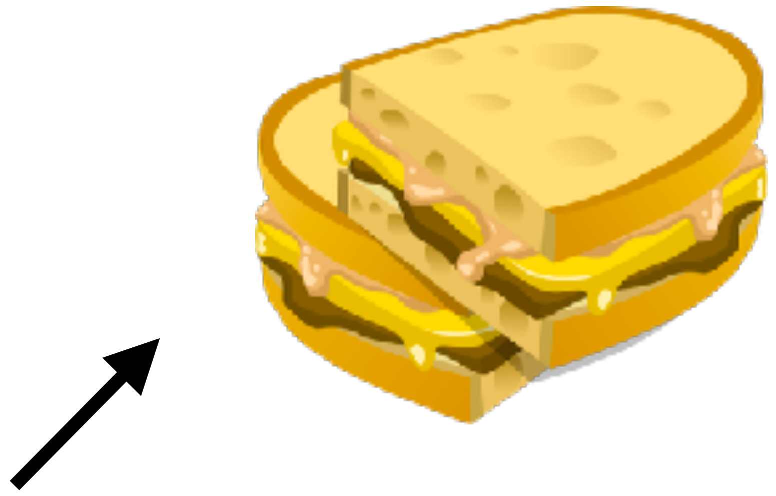
From Concepts to Functions

Concept Learning as Classification

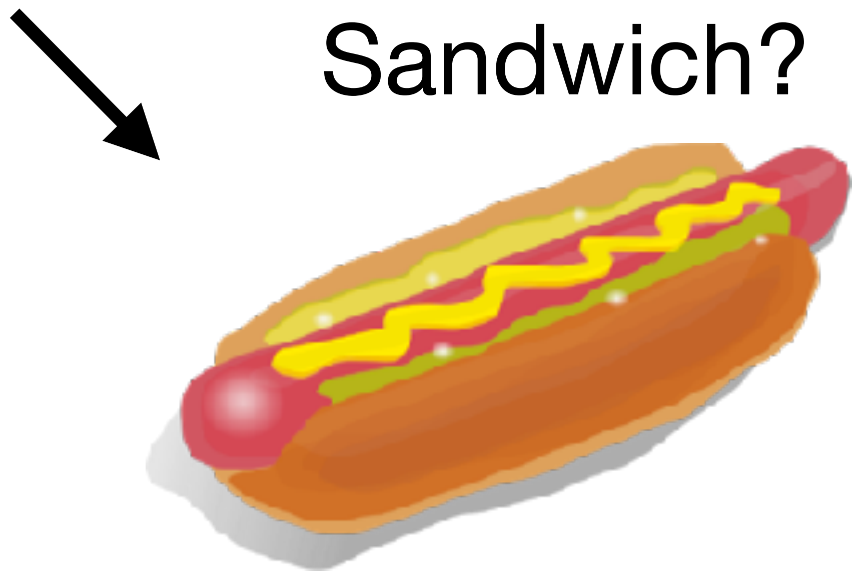
Previous Experiences



Sandwich!

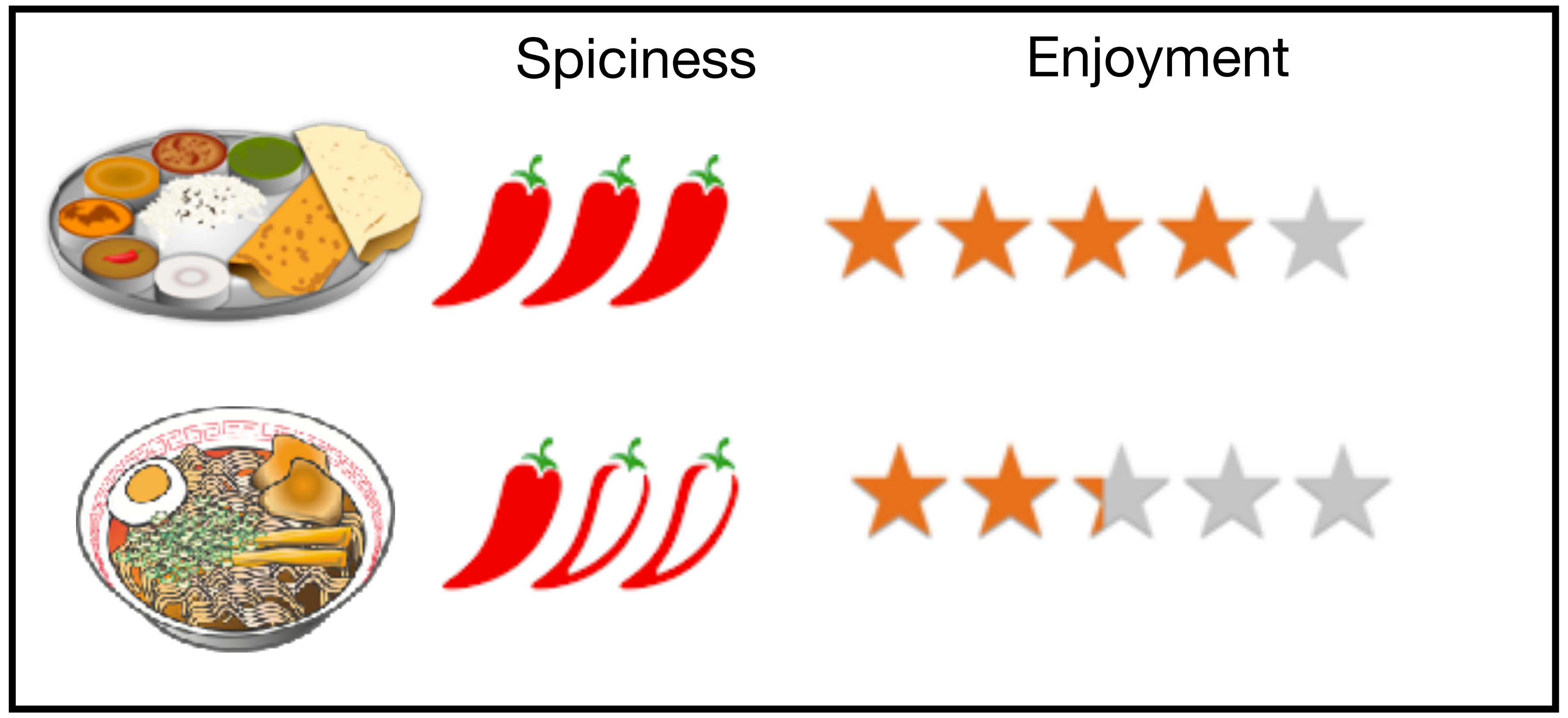


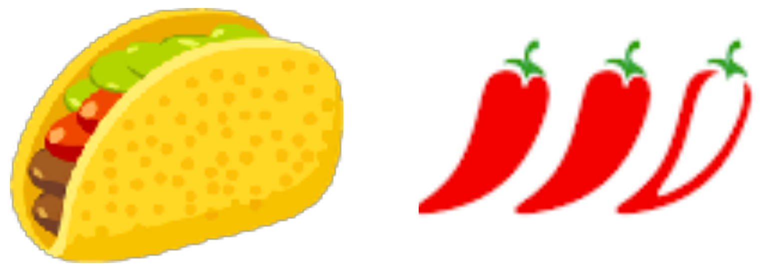
Sandwich?



Function learning as Regression

Previous Experiences





?

Today's agenda

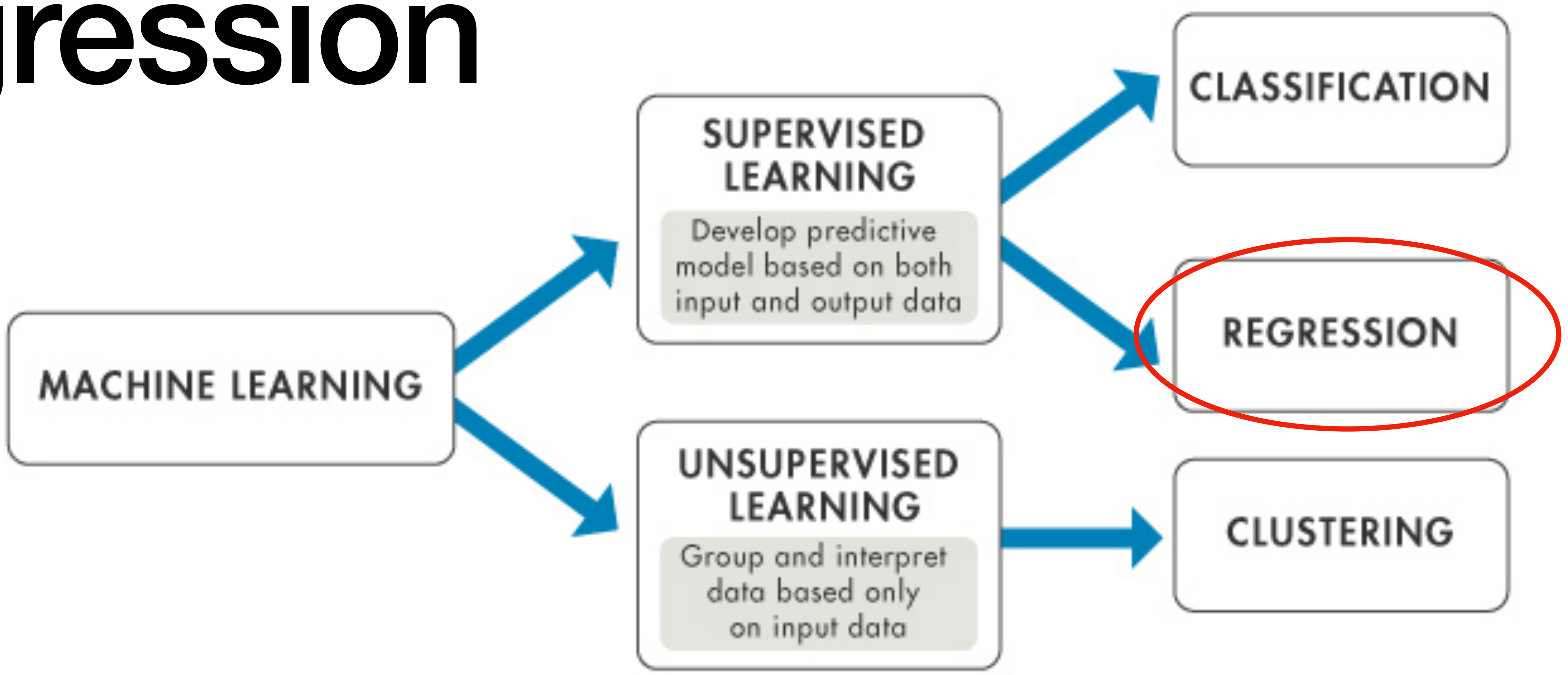
- Early Psychological research on how people learn explicit functions
  - Rule-based
  - Similarity-based
  - Hybrid using Bayesian function learning
- Implicit function learning as a key part of generalization in RL
- Modeling human generalization and exploration in RL
  - Spatially correlated bandit (Wu et al,. 2018; Giron et al., 2023)
  - Generalization to abstract (Wu et al., 2020) and graph-structured domains (Wu et al,. 2021)
  - Open challenges

Function learning as regression

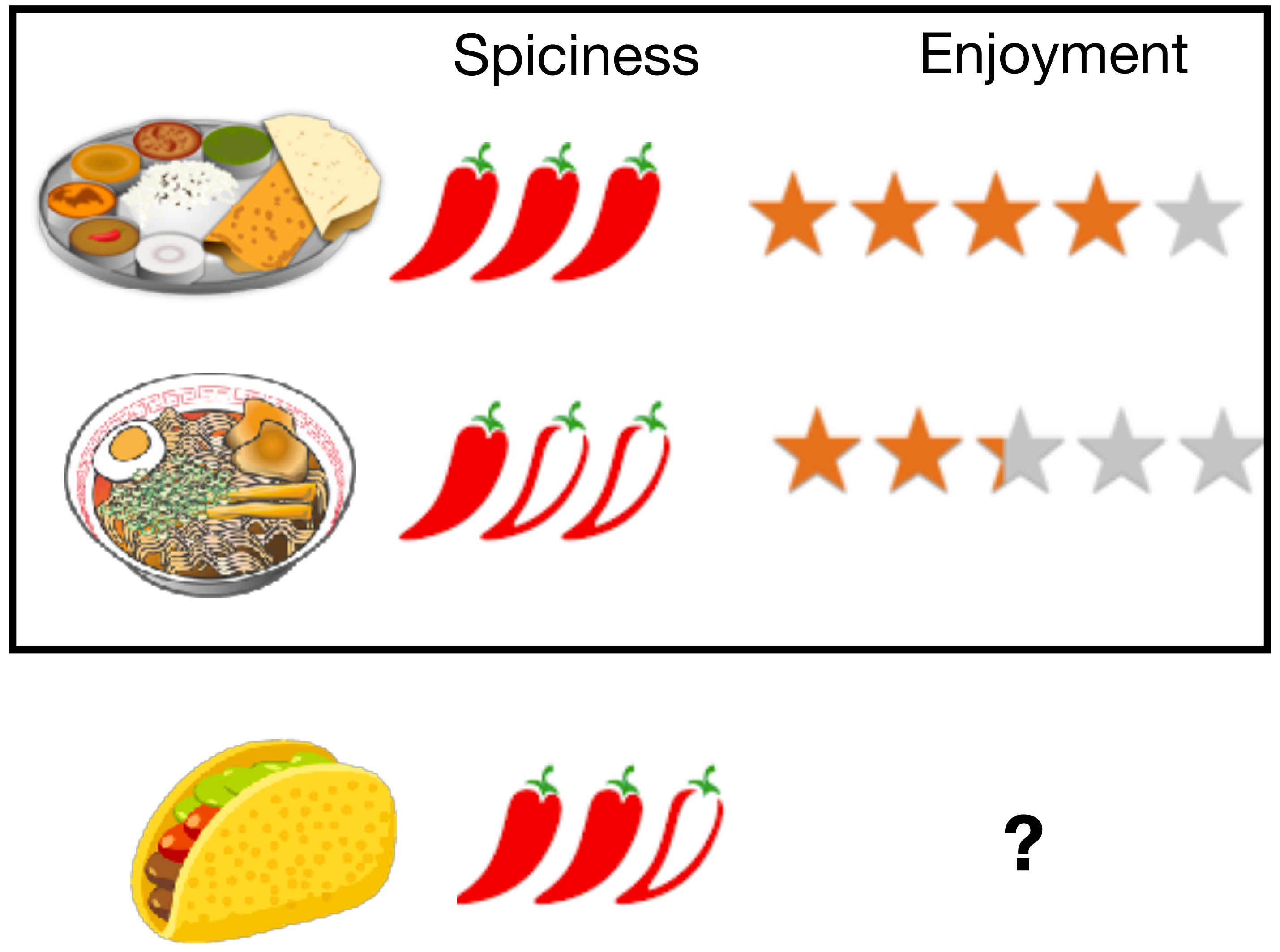
- **Regression** is that other branch of supervised learning problems we previously skipped over
- Rather than predicting *discrete* categories, we want to learn to predict a *continuous* real-valued variable
    - Learning a function mapping input space X to target variable Y

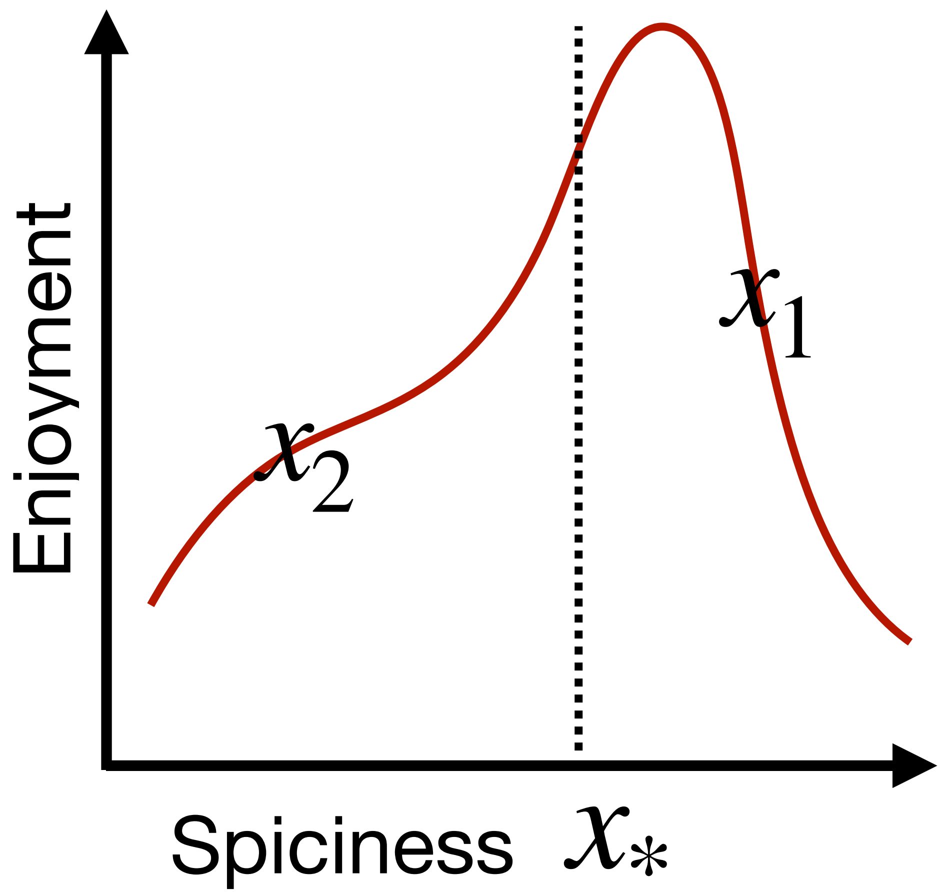
$$f : X \rightarrow Y \text{ where } y = f(x)$$

- To make a prediction about so new situation x_*, we simply evaluate the function: $y_* = f(x_*)$
- *But how do we learn this function*? For any set of datapoints, there are an infinite number of functions that pass through them



Previous Experiences

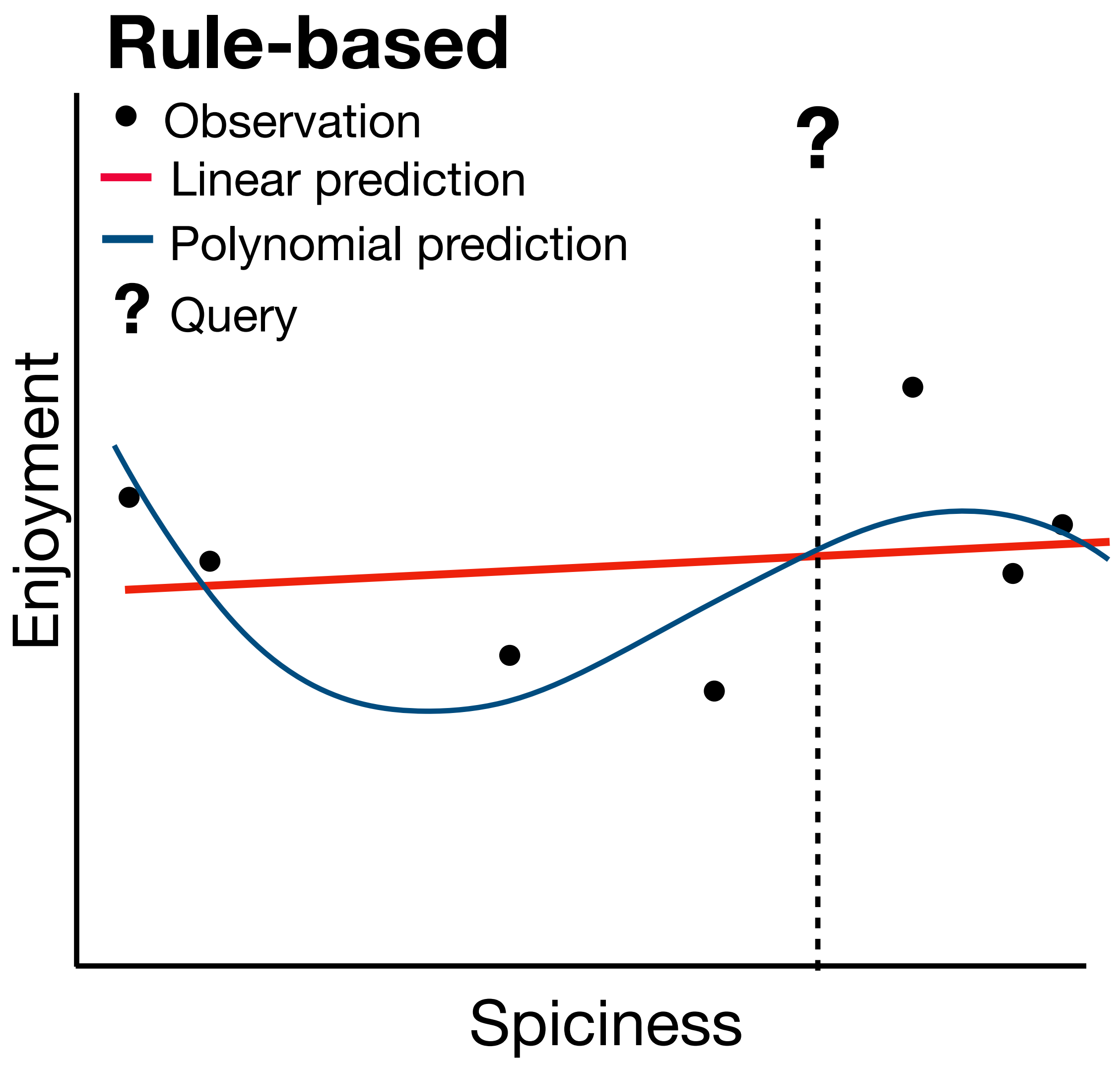




Theories of Function Learning

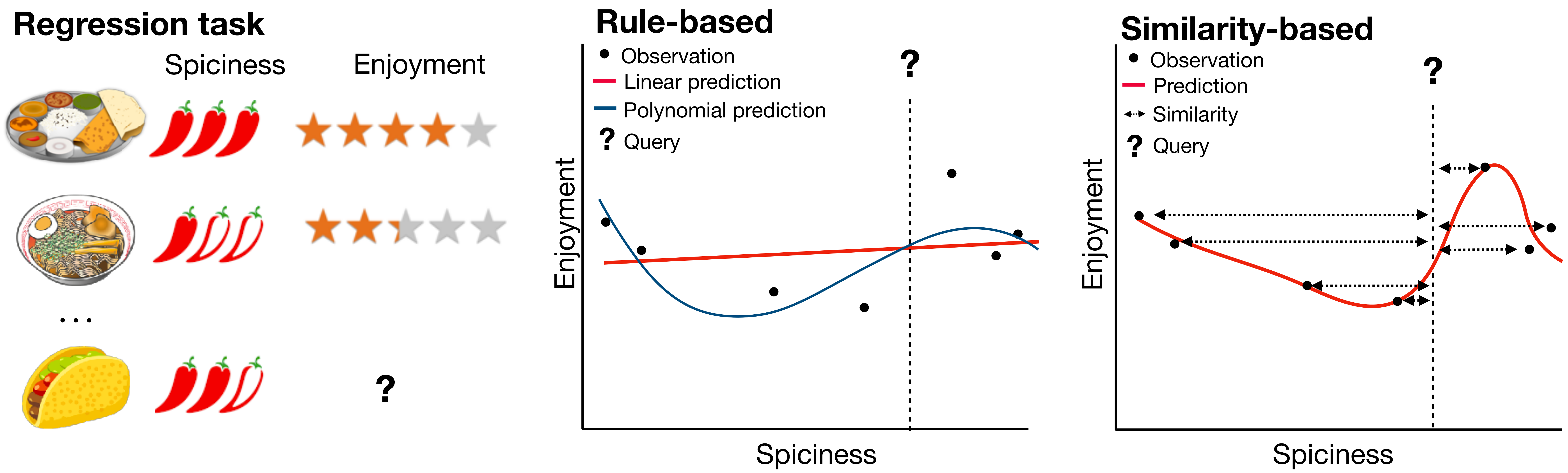
Regression task

Theories of Function Learning



- *Rules* describe an explicit parametric family of candidate functions (e.g., linear or polynomial)
  (Carroll, 1963; Brehmer, 1976)

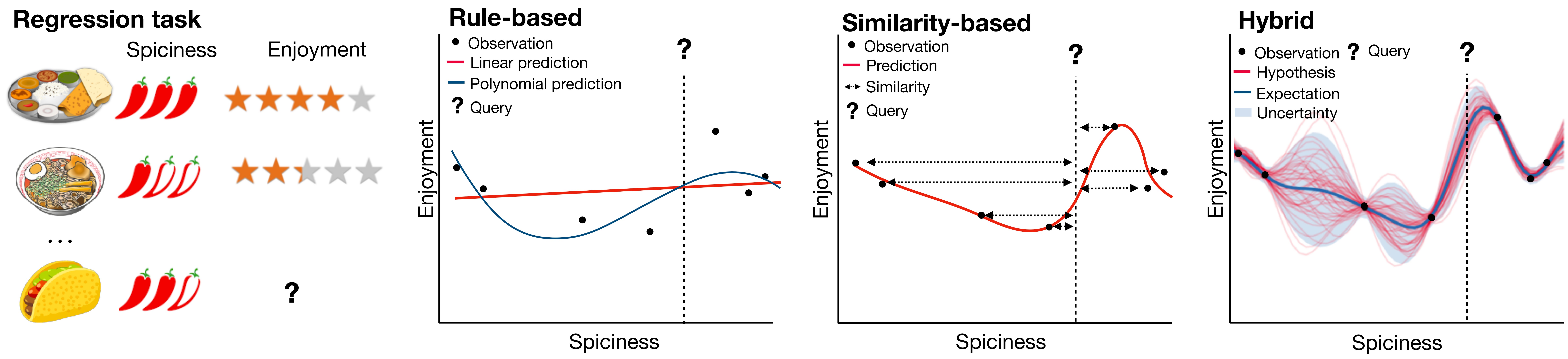
Theories of Function Learning



- *Rules* describe an explicit parametric family of candidate functions (e.g., linear or polynomial)
  (Carroll, 1963; Brehmer, 1976)

- *Similarity* uses the generic principle that similar inputs produce similar outputs (often learned using ANNs) as the basis of generalization
  (McClelland et al., 1986; Busemeyer et al., 1997)

Theories of Function Learning



- *Rules* describe an explicit parametric family of candidate functions (e.g., linear or polynomial)
  (Carroll, 1963; Brehmer, 1976)

- *Similarity* uses the generic principle that similar inputs produce similar outputs (often learned using ANNs) as the basis of generalization
  (McClelland et al., 1986; Busemeyer et al., 1997)

- *Hybrids* combine elements of both: Gaussian process (GP) regression uses kernel similarity to learn a distribution over functions, and can compositionally combine kernels like we can combine multiple rules
  (Rasmussen & Williams, 2005; Mercer, PhilTransRoySoc 1909; Lucas et al., PBR 2015)

Rule-based theories of function learning

- Carroll (1963) was one of the first to study how people learned continuous mappings between stimuli and responses
  - Rather than learning discrete S-R associations, people learn functions
  - Functions are not just a response, but correspond to a set of rules or programs, allowing for interpolation and extrapolation
- Experiment using relationships such as $y = 1.22x + 1.0$ or $y = -5.1x + 0.2x^2 + 32.60$

Rule-based theories of function learning

- Carroll (1963) was one of the first to study how people learned continuous mappings between stimuli and responses
  - Rather than learning discrete S-R associations, people learn functions
  - Functions are not just a response, but correspond to a set of rules or programs, allowing for interpolation and extrapolation
- Experiment using relationships such as $y = 1.22x + 1.0$ or $y = -5.1x + 0.2x^2 + 32.60$

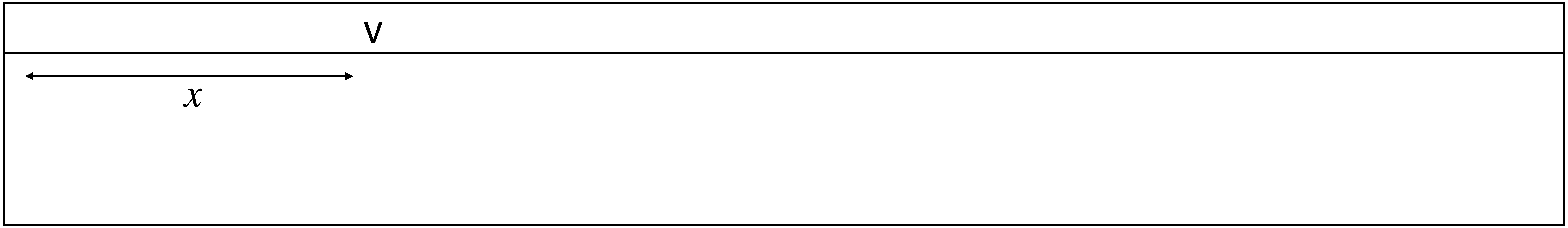
stimuli

V

Rule-based theories of function learning

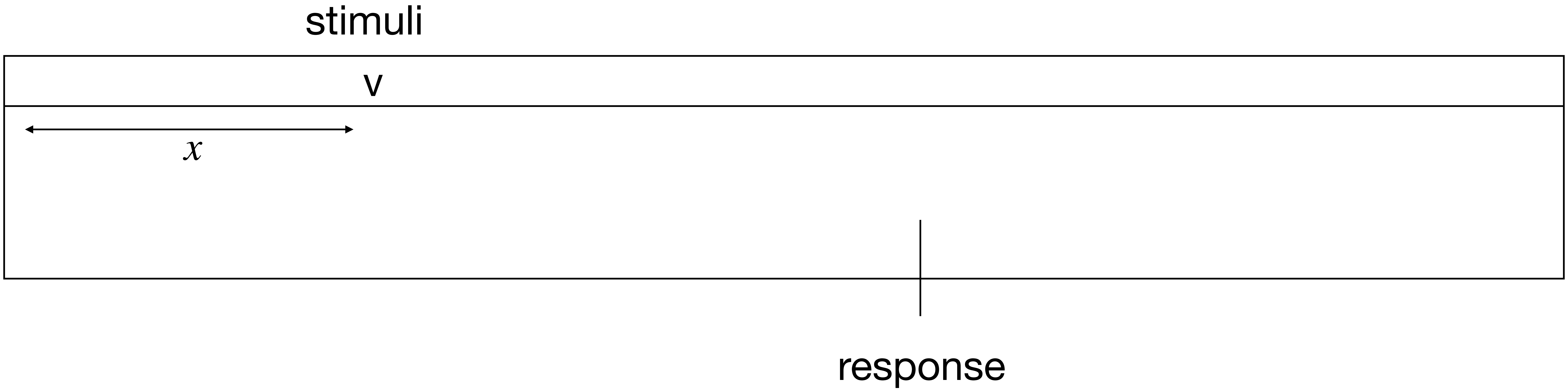
- Carroll (1963) was one of the first to study how people learned continuous mappings between stimuli and responses
  - Rather than learning discrete S-R associations, people learn functions
  - Functions are not just a response, but correspond to a set of rules or programs, allowing for interpolation and extrapolation
- Experiment using relationships such as $y = 1.22x + 1.0$ or $y = -5.1x + 0.2x^2 + 32.60$

stimuli



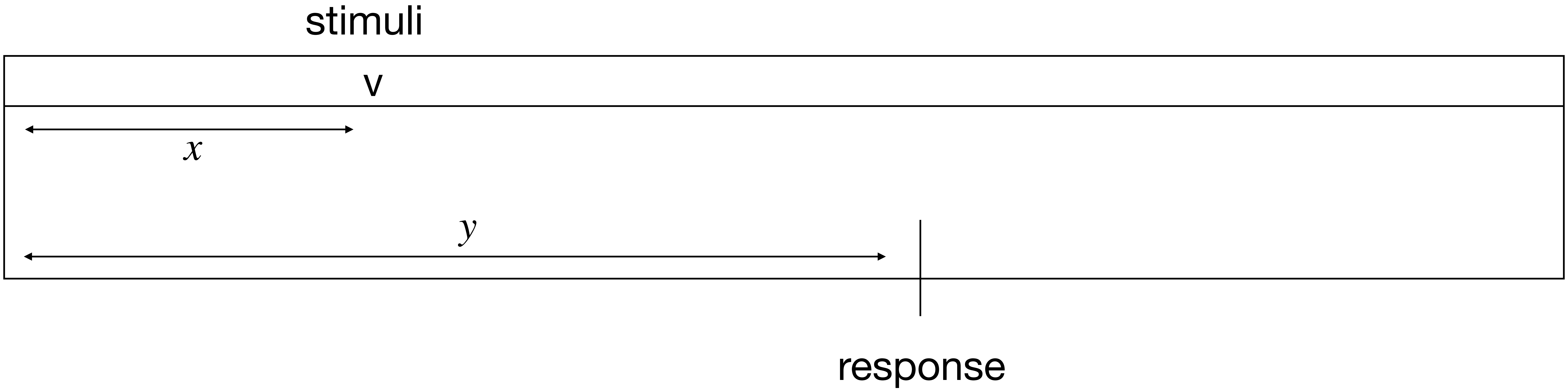
Rule-based theories of function learning

- Carroll (1963) was one of the first to study how people learned continuous mappings between stimuli and responses
  - Rather than learning discrete S-R associations, people learn functions
  - Functions are not just a response, but correspond to a set of rules or programs, allowing for interpolation and extrapolation
- Experiment using relationships such as $y = 1.22x + 1.0$ or $y = -5.1x + 0.2x^2 + 32.60$



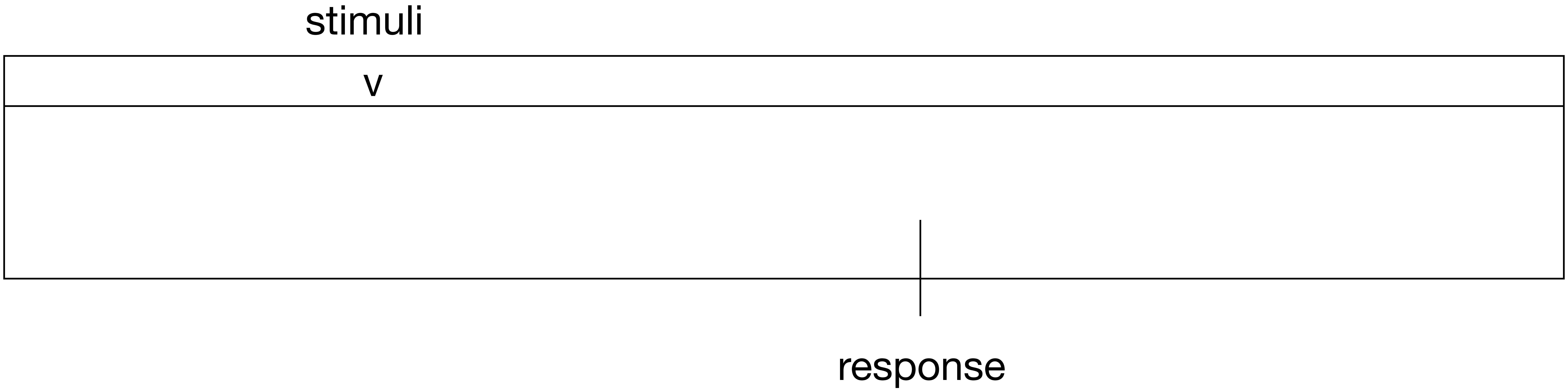
Rule-based theories of function learning

- Carroll (1963) was one of the first to study how people learned continuous mappings between stimuli and responses
  - Rather than learning discrete S-R associations, people learn functions
  - Functions are not just a response, but correspond to a set of rules or programs, allowing for interpolation and extrapolation
- Experiment using relationships such as $y = 1.22x + 1.0$ or $y = -5.1x + 0.2x^2 + 32.60$



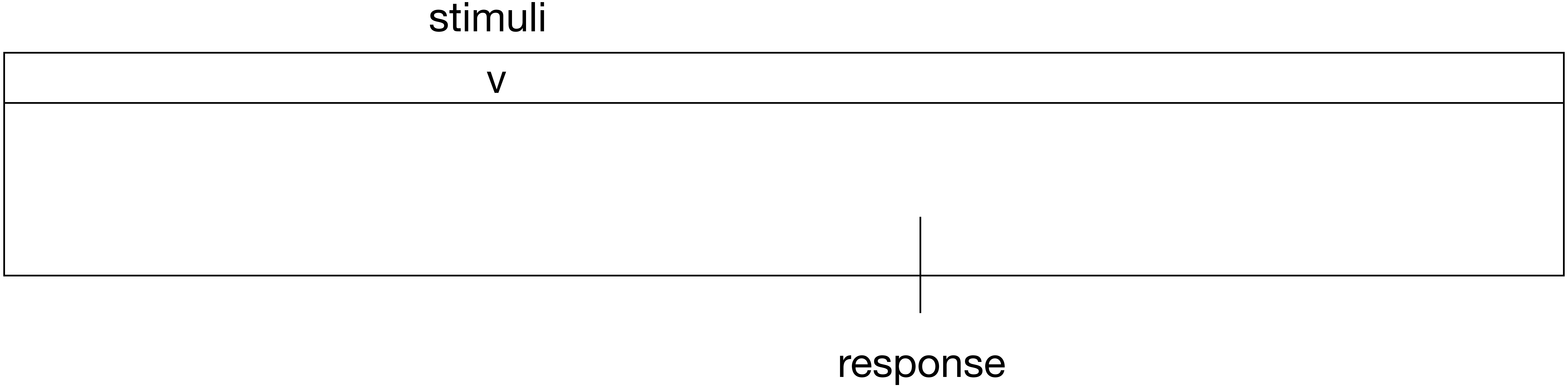
Rule-based theories of function learning

- Carroll (1963) was one of the first to study how people learned continuous mappings between stimuli and responses
  - Rather than learning discrete S-R associations, people learn functions
  - Functions are not just a response, but correspond to a set of rules or programs, allowing for interpolation and extrapolation
- Experiment using relationships such as $y = 1.22x + 1.0$ or $y = -5.1x + 0.2x^2 + 32.60$



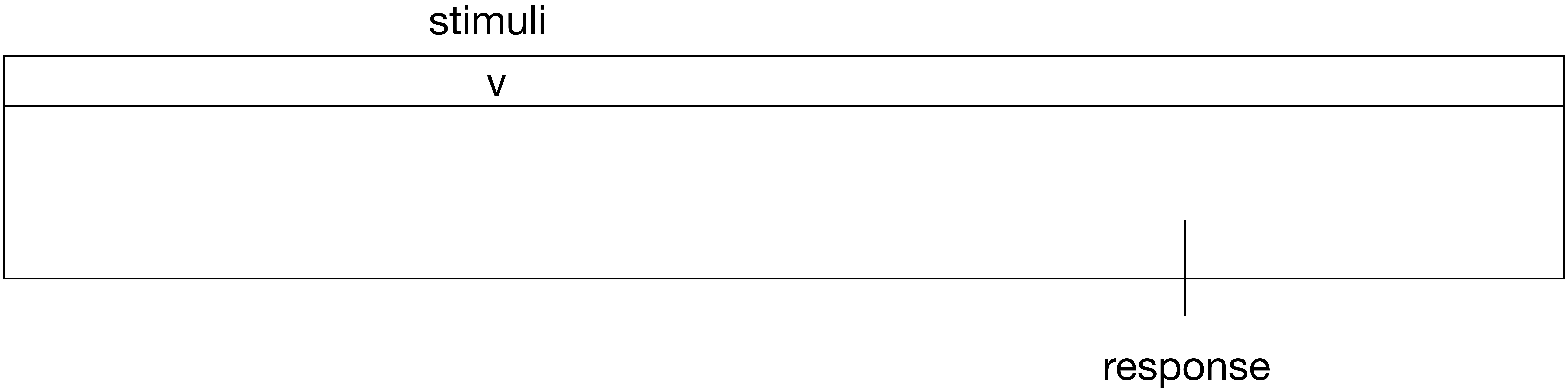
Rule-based theories of function learning

- Carroll (1963) was one of the first to study how people learned continuous mappings between stimuli and responses
  - Rather than learning discrete S-R associations, people learn functions
  - Functions are not just a response, but correspond to a set of rules or programs, allowing for interpolation and extrapolation
- Experiment using relationships such as $y = 1.22x + 1.0$ or $y = -5.1x + 0.2x^2 + 32.60$



Rule-based theories of function learning

- Carroll (1963) was one of the first to study how people learned continuous mappings between stimuli and responses
  - Rather than learning discrete S-R associations, people learn functions
  - Functions are not just a response, but correspond to a set of rules or programs, allowing for interpolation and extrapolation
- Experiment using relationships such as $y = 1.22x + 1.0$ or $y = -5.1x + 0.2x^2 + 32.60$



Results and interpretation

- Participants were shown arbitrary relationships between x and y in the training regime
- … their responses showed that they learned functions rather than just discrete associations, based on ability to *interprolate* and *extrapolate*
- In general, participants had an inductive biases for simpler functions (e.g., lower degree polynomial)
- Early **rule-based theories** assumed people learn functions by estimating the parameters for a class of functions (e.g., polynomials) using a process equivalent to regression
  - The class of function corresponds to a hypothesized **rule** about the relationship between variables
  - e.g., the law of gravity: $F = G\frac{m_1 m_2}{r}$

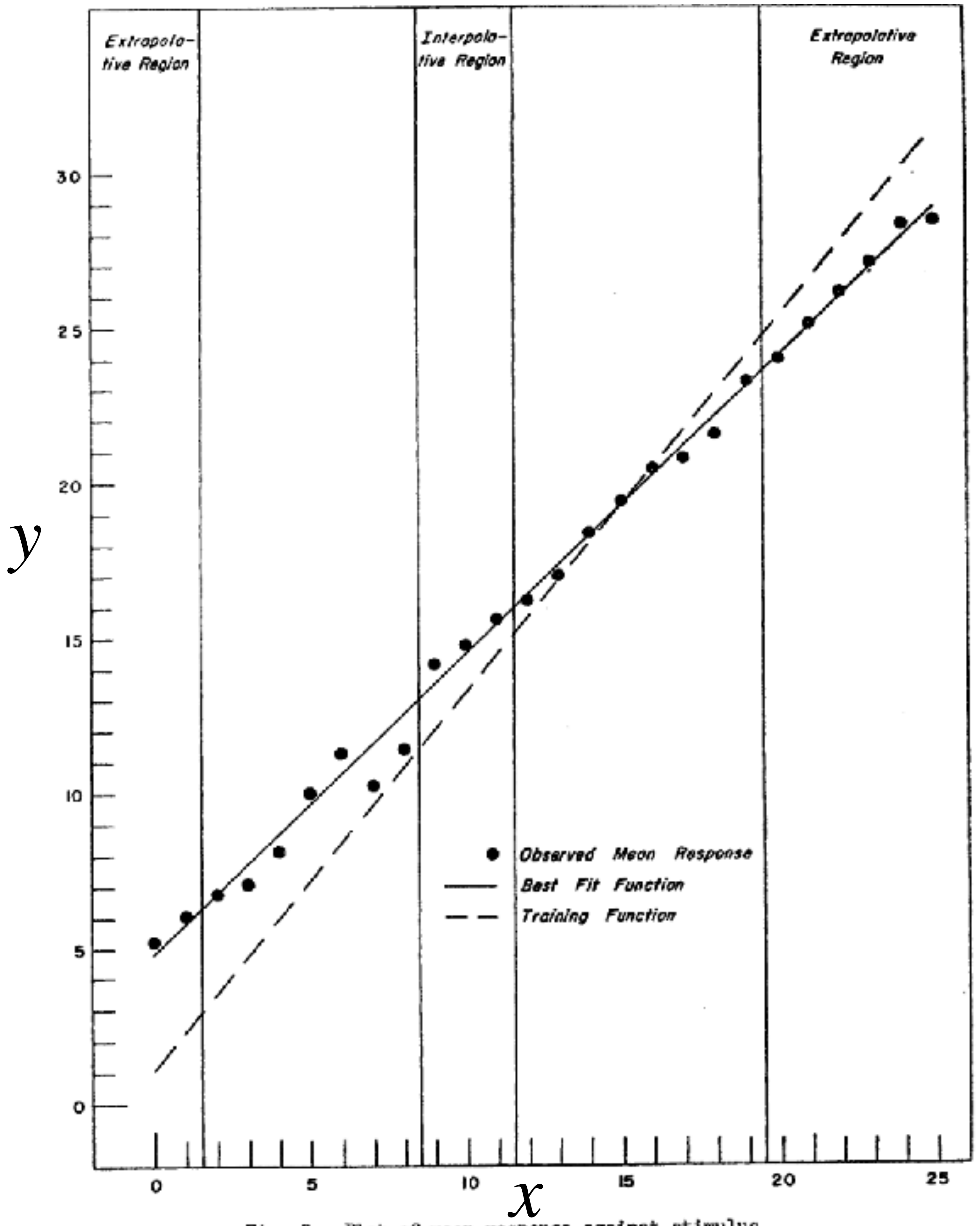


Fig. 3. Plot of mean response against stimulus for subject #4, condition I.

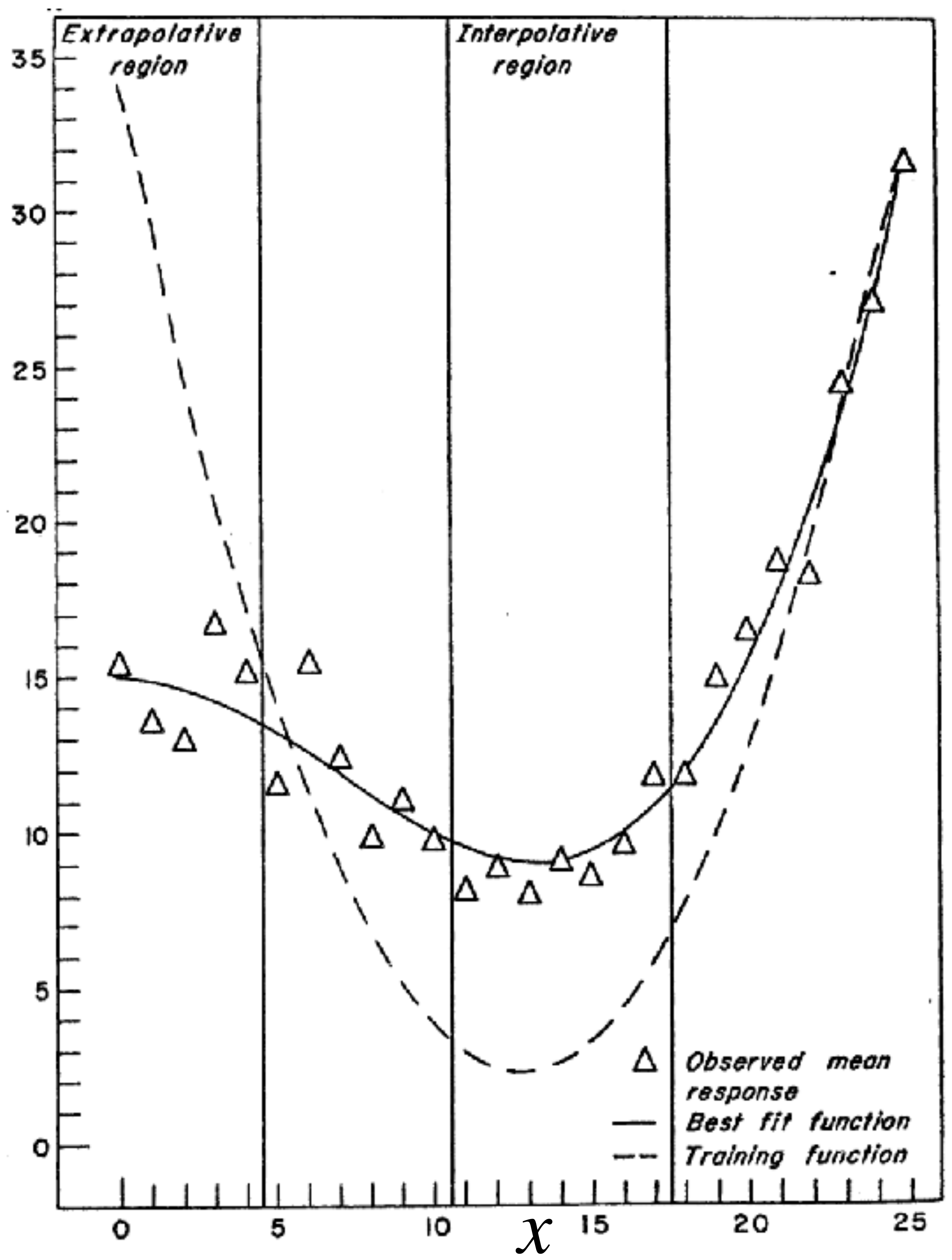


Fig. 4. Plot of mean response against stimulus for subject #20, condition III.

Results and interpretation

- Participants were shown arbitrary relationships between x and y in the training regime
- … their responses showed that they learned functions rather than just discrete associations, based on ability to *interprolate* and *extrapolate*
- In general, participants had an inductive biases for simpler functions (e.g., lower degree polynomial)
- Early **rule-based theories** assumed people learn functions by estimating the parameters for a class of functions (e.g., polynomials) using a process equivalent to regression
    - The class of function corresponds to a hypothesized **rule** about the relationship between variables
    - e.g., the law of gravity: $F = G\frac{m_1 m_2}{r}$

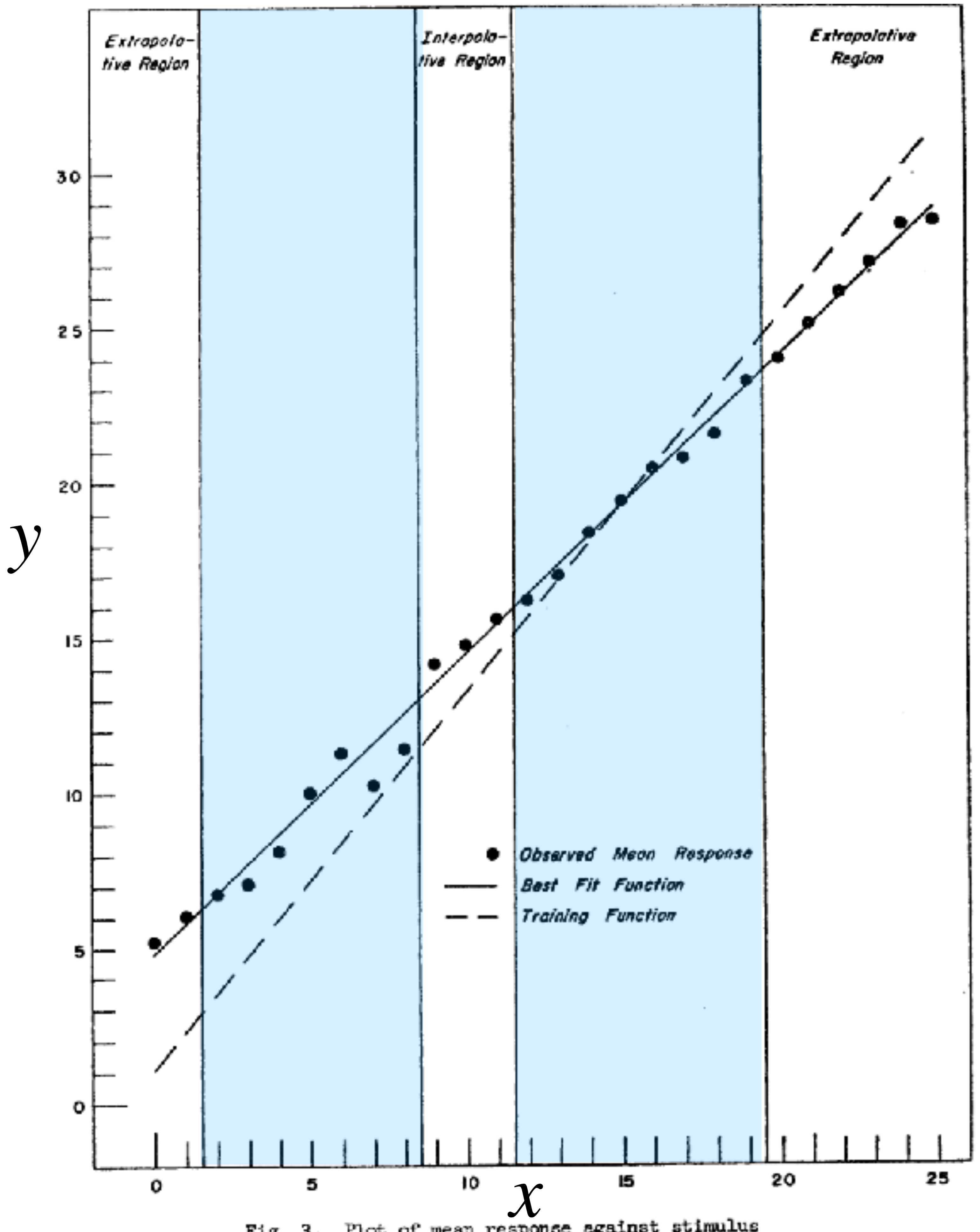


Fig. 3. Plot of mean response against stimulus for subject #4, condition I.

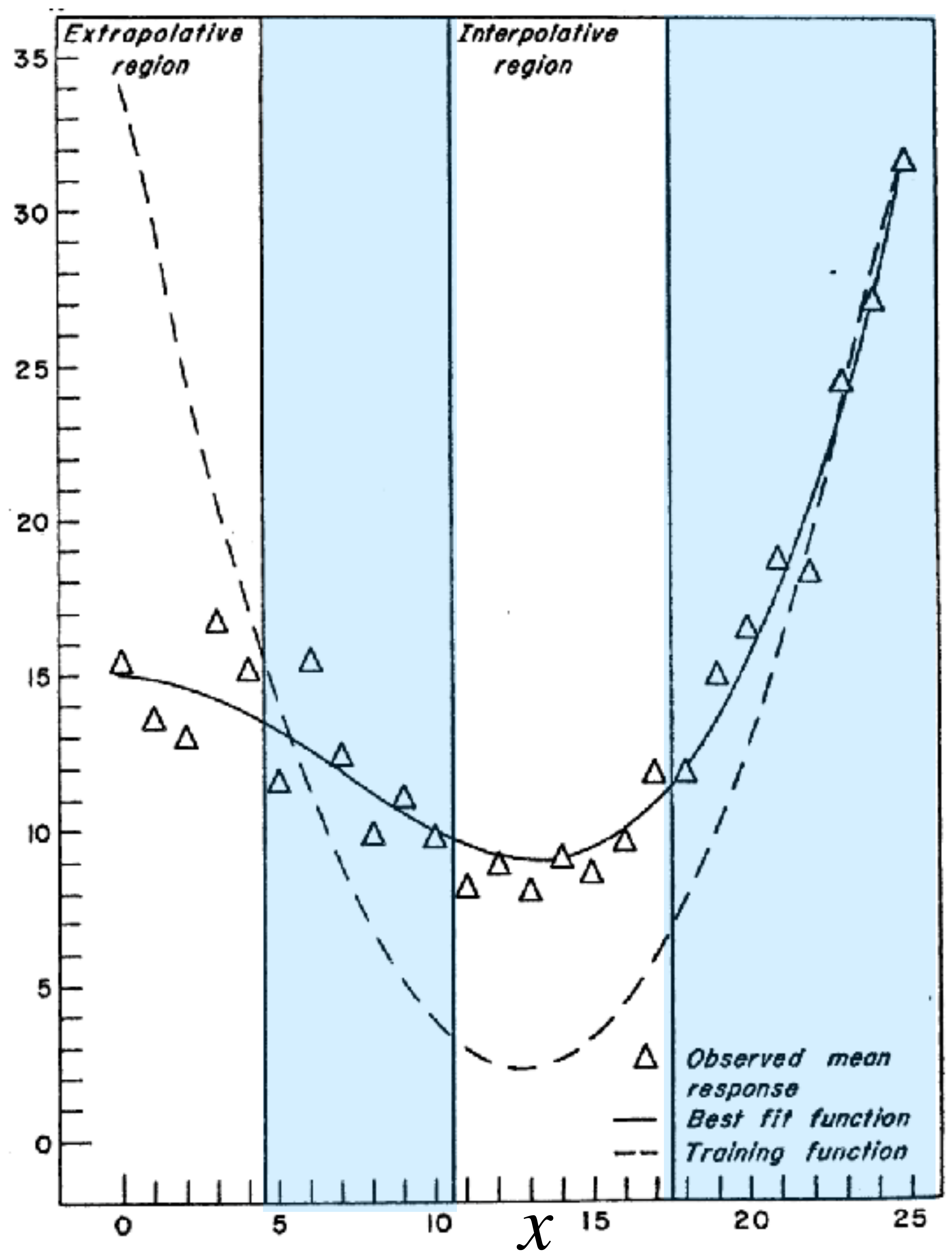


Fig. 4. Plot of mean response against stimulus for subject #20, condition III.

Results and interpretation

- Participants were shown arbitrary relationships between x and y in the training regime
- … their responses showed that they learned functions rather than just discrete associations, based on ability to *interprolate* and *extrapolate*
- In general, participants had an inductive biases for simpler functions (e.g., lower degree polynomial)
- Early **rule-based theories** assumed people learn functions by estimating the parameters for a class of functions (e.g., polynomials) using a process equivalent to regression
  - The class of function corresponds to a hypothesized **rule** about the relationship between variables
  - e.g., the law of gravity: $F = G\frac{m_1 m_2}{r}$

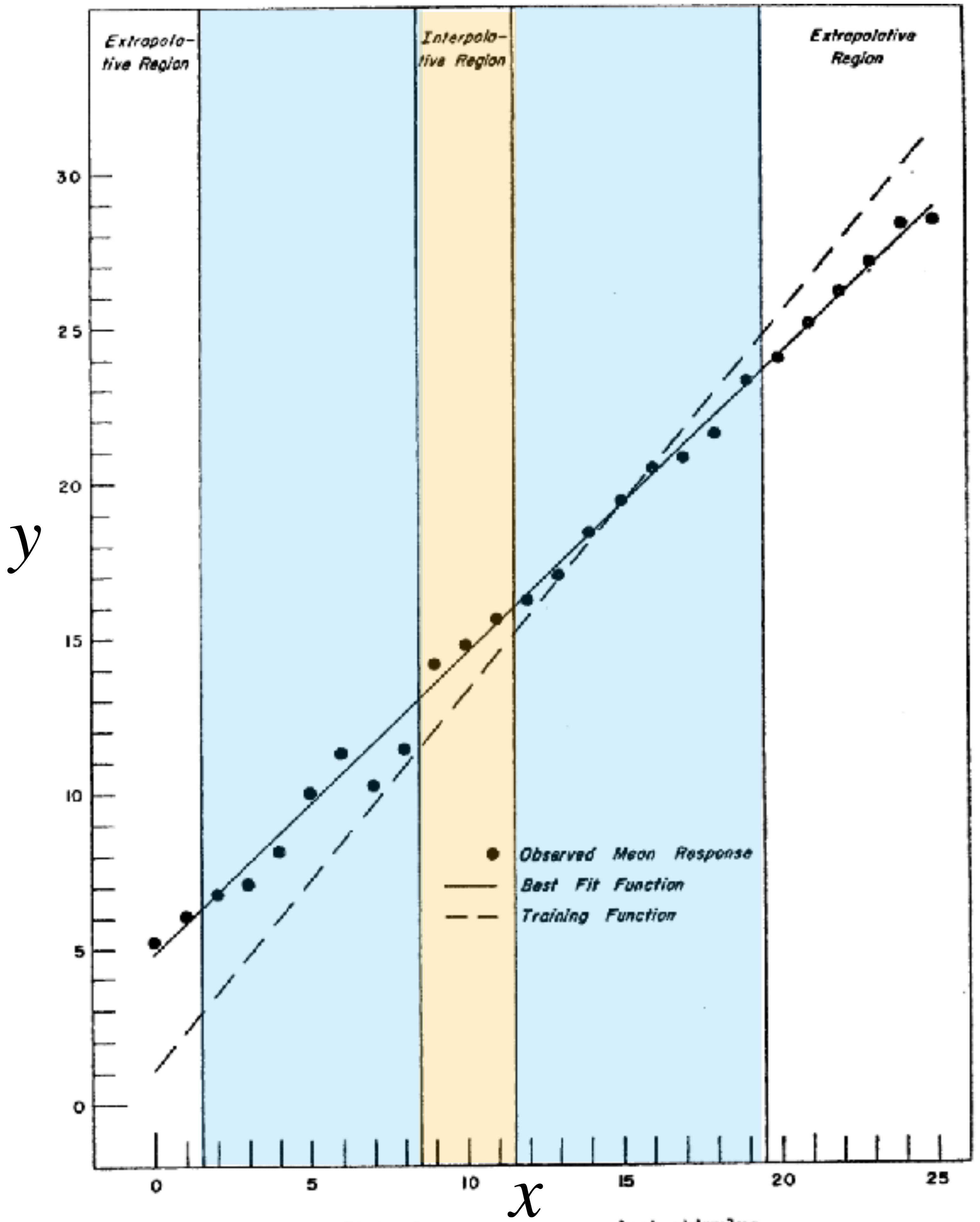


Fig. 3. Plot of mean response against stimulus for subject #4, condition I.

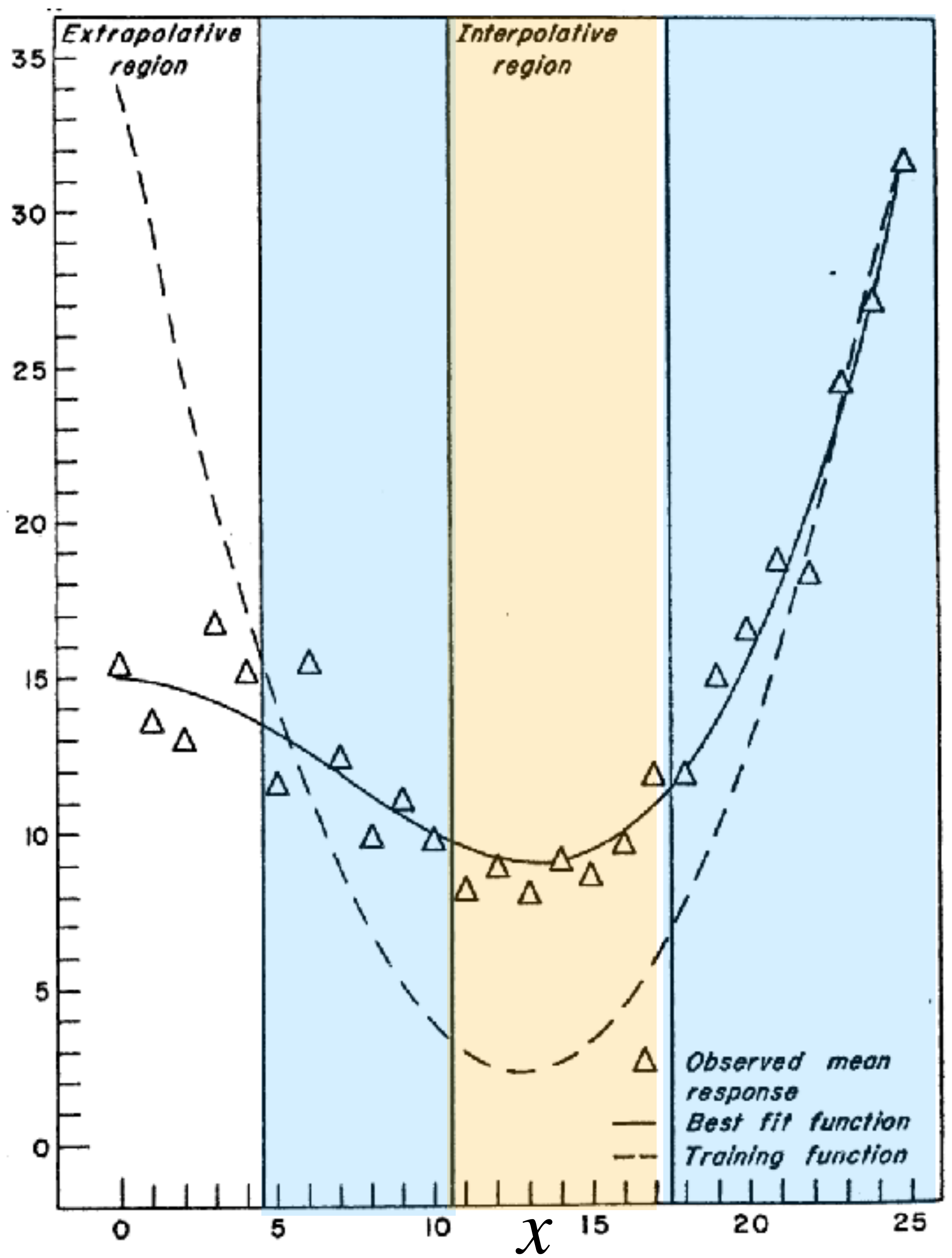


Fig. 4. Plot of mean response against stimulus for subject #20, condition III.

Results and interpretation

- Participants were shown arbitrary relationships between x and y in the training regime
- … their responses showed that they learned functions rather than just discrete associations, based on ability to *interprolate* and *extrapolate*
- In general, participants had an inductive biases for simpler functions (e.g., lower degree polynomial)
- Early **rule-based theories** assumed people learn functions by estimating the parameters for a class of functions (e.g., polynomials) using a process equivalent to regression
  - The class of function corresponds to a hypothesized **rule** about the relationship between variables
  - e.g., the law of gravity: $F = G\frac{m_1 m_2}{r}$

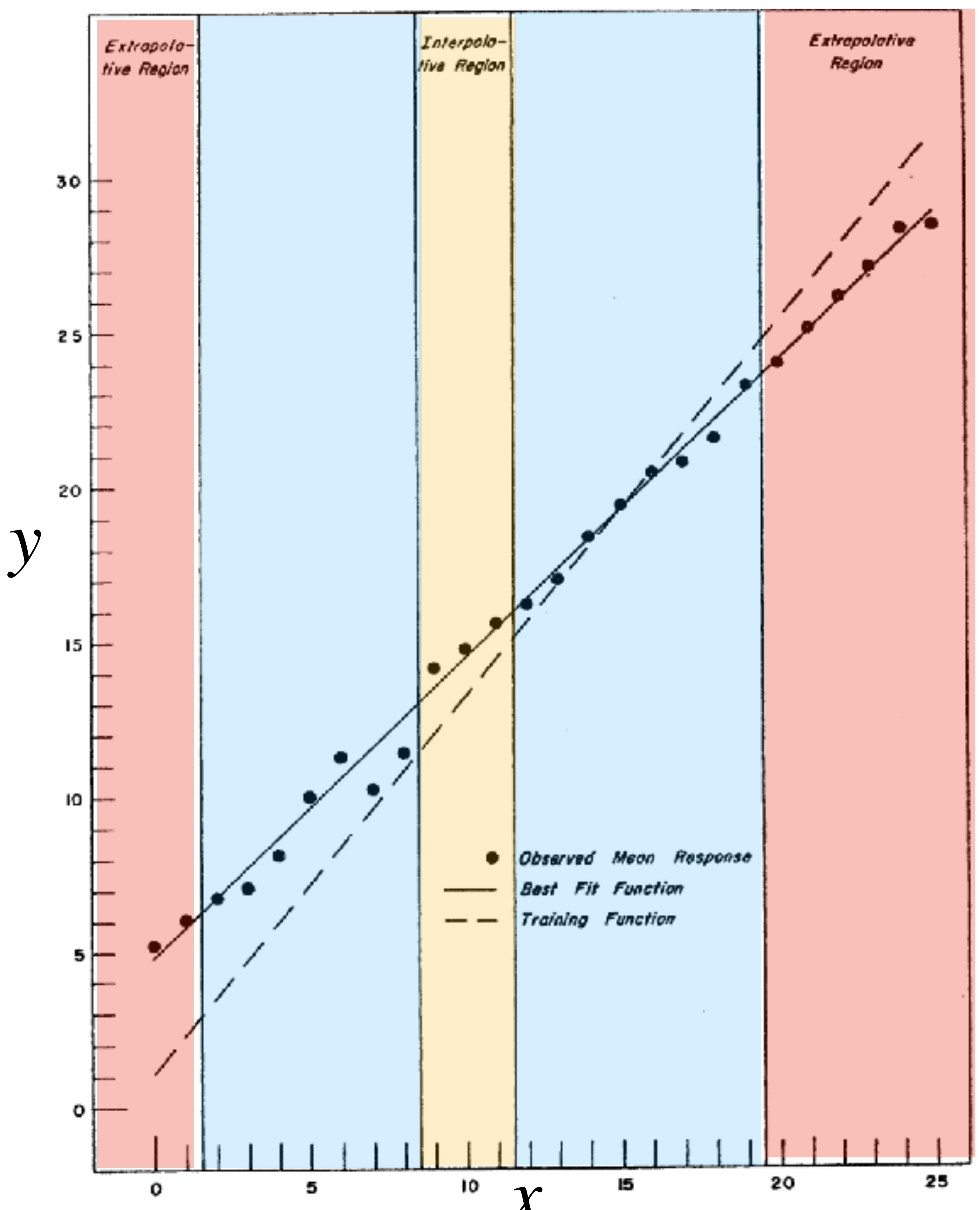


Fig. 3. Plot of mean response against stimulus for subject #4, condition I.

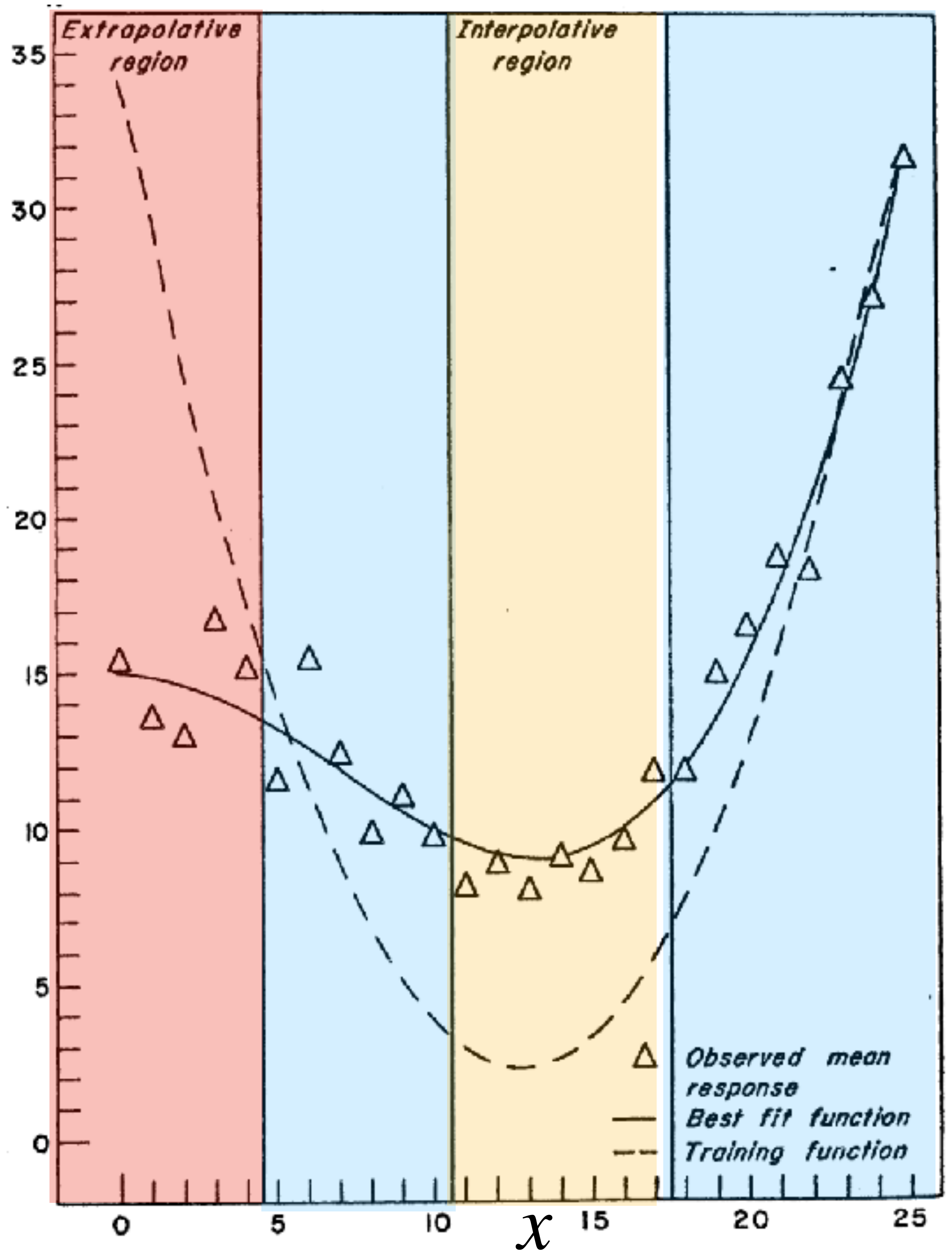
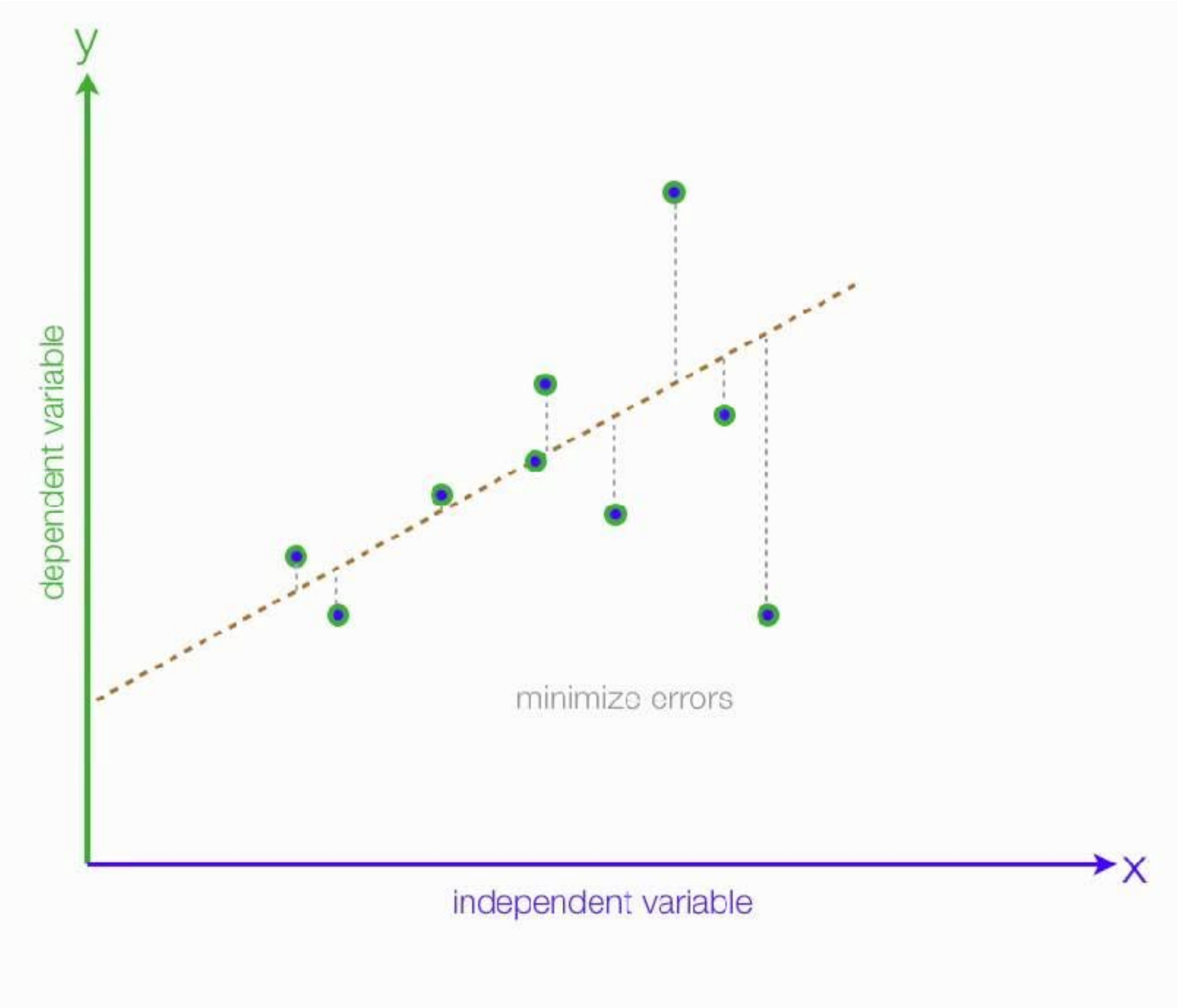


Fig. 4. Plot of mean response against stimulus for subject #20, condition III.

Linear regression

- *Find a line that minimizes errors*

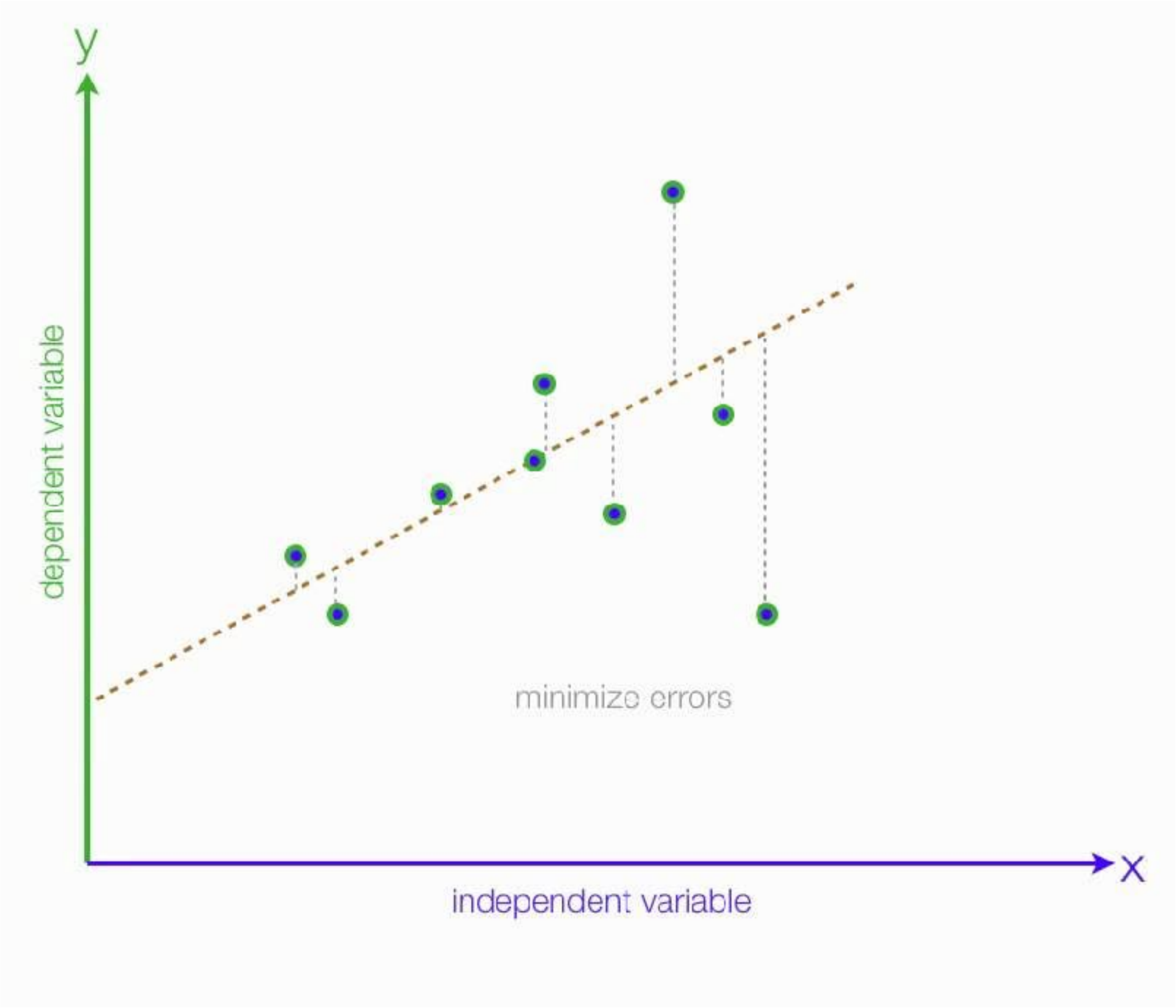


Linear regression

- *Find a line that minimizes errors*
- How you learn it in high school:

$$y = mx + b \quad \longleftarrow \text{intercept}$$

slope ↗



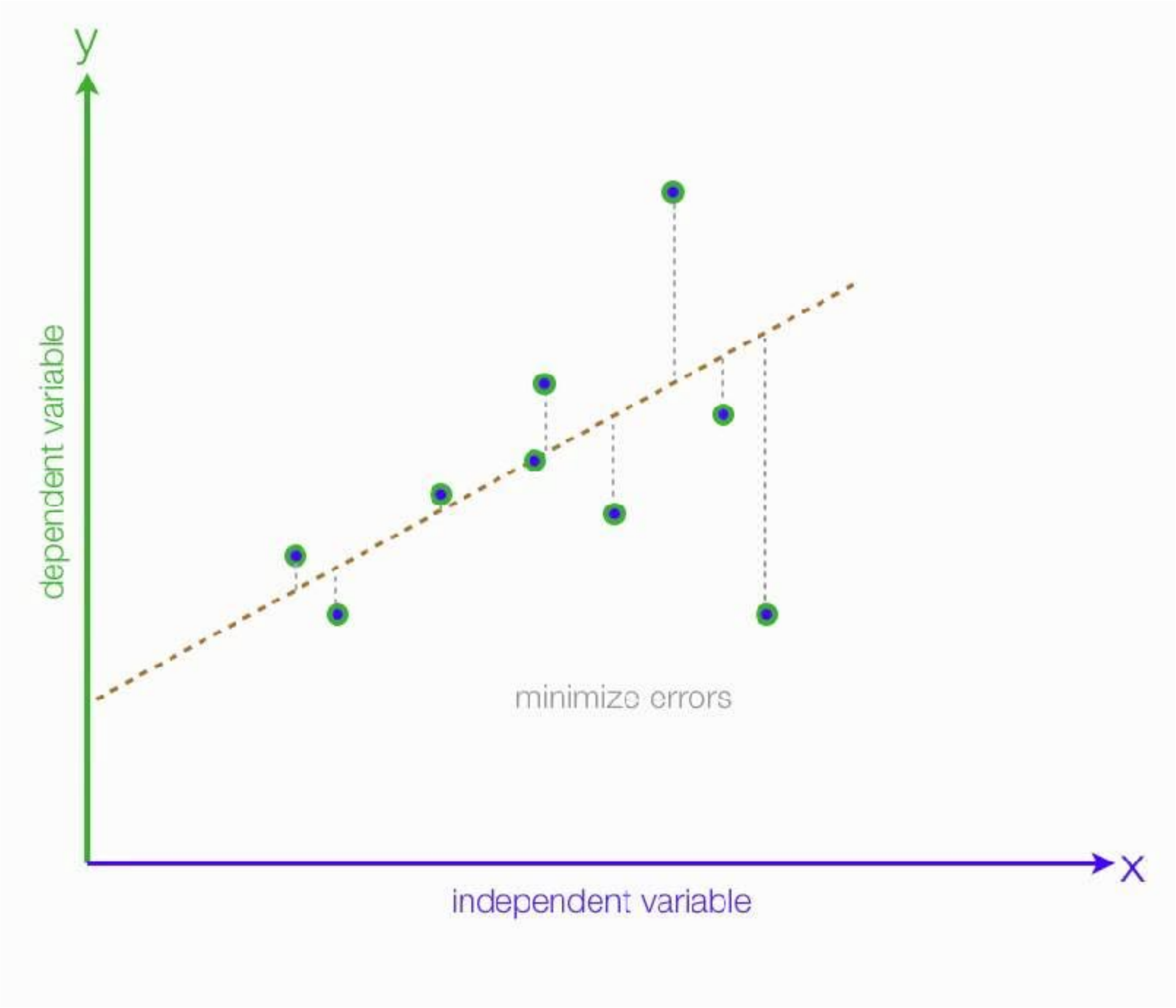
Linear regression

- *Find a line that minimizes errors*
- How you learn it in high school:

  $y = mx + b$ ⟵ intercept

  slope ↗
- Linear algebra version:

  $y = X^\top \mathbf{w} + \epsilon$
  - X is a matrix of the data $[\mathbf{x}_1, \ldots \mathbf{x}_n]$
  - We append $x_{i,0} = 1$ to each $\mathbf{x}_i$ vector to account for the intercept
  - $\mathbf{w}$ are the *weights*
  - $\epsilon \sim \mathcal{N}(0, \sigma_n^2)$ is i.i.d. noise



Linear regression

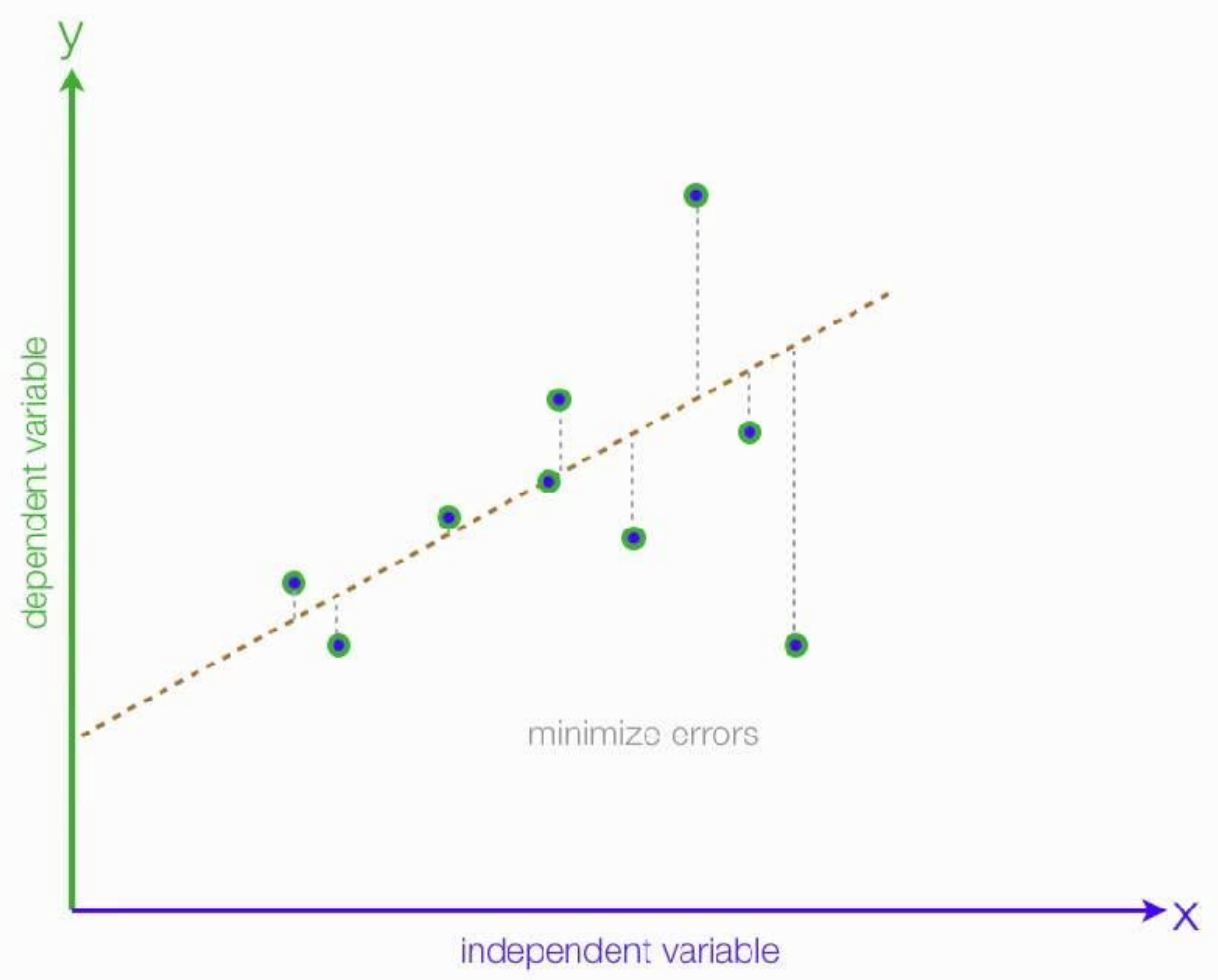
- *Find a line that minimizes errors*
- How you learn it in high school:

  $y = mx + b$ ⟵ intercept

  slope ↗
- Linear algebra version:

  $y = X^{\top}\mathbf{w} + \epsilon$

  - X is a matrix of the data $[\mathbf{x}_1, \ldots \mathbf{x}_n]$
  - We append $x_{i,0} = 1$ to each $\mathbf{x}_i$ vector to account for the intercept
  - $\mathbf{w}$ are the *weights*
  - $\epsilon \sim \mathcal{N}(0, \sigma_n^2)$ is i.i.d. noise



Maximum Likelihood Estimation (MLE)

- MLE of weights can be found by minimizing the Residual Sum of Squares (RSS):

$$RSS(\mathbf{w}) = \sum_{i}^{n} (y_i - \hat{y}_i)^2 = \|\mathbf{y} - \mathbf{X}^{\top}\mathbf{w}\|^2$$

- An analytic solution is available through the Moore-Penrose psuedoinverse (Penrose, 1955): $\hat{\mathbf{w}} = \left(\mathbf{X}^{\top}\mathbf{X}\right)^{-1}\mathbf{X}^{\top}\mathbf{y}$

Linear regression

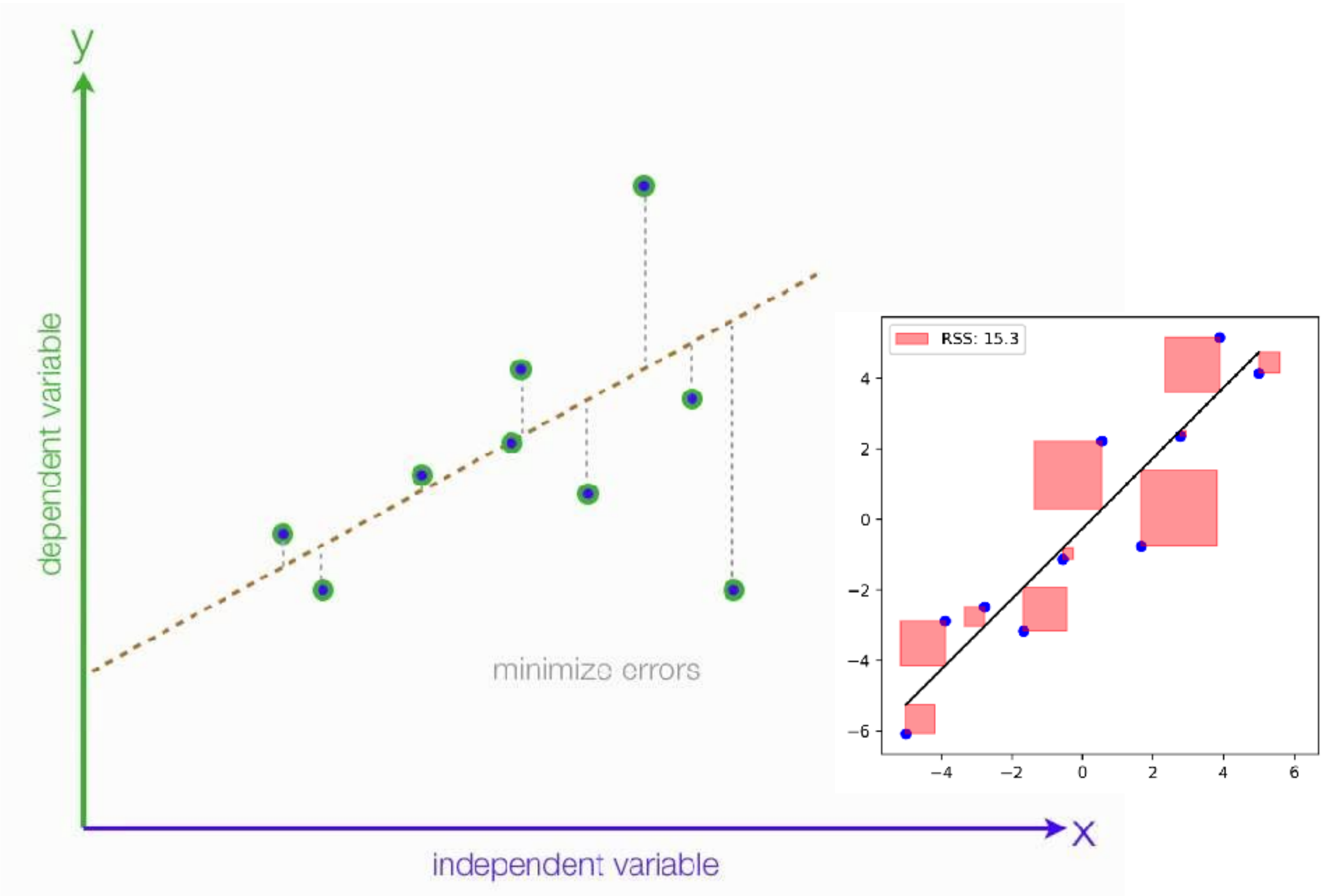
- *Find a line that minimizes errors*
- How you learn it in high school:

  $y = mx + b$ ⟵ intercept

  slope ↗
- Linear algebra version:

  $y = X^{\top}\mathbf{w} + \epsilon$

  - X is a matrix of the data $[\mathbf{x}_1, \ldots \mathbf{x}_n]$
  - We append $x_{i,0} = 1$ to each $\mathbf{x}_i$ vector to account for the intercept
  - $\mathbf{w}$ are the *weights*
  - $\epsilon \sim \mathcal{N}(0, \sigma_n^2)$ is i.i.d. noise



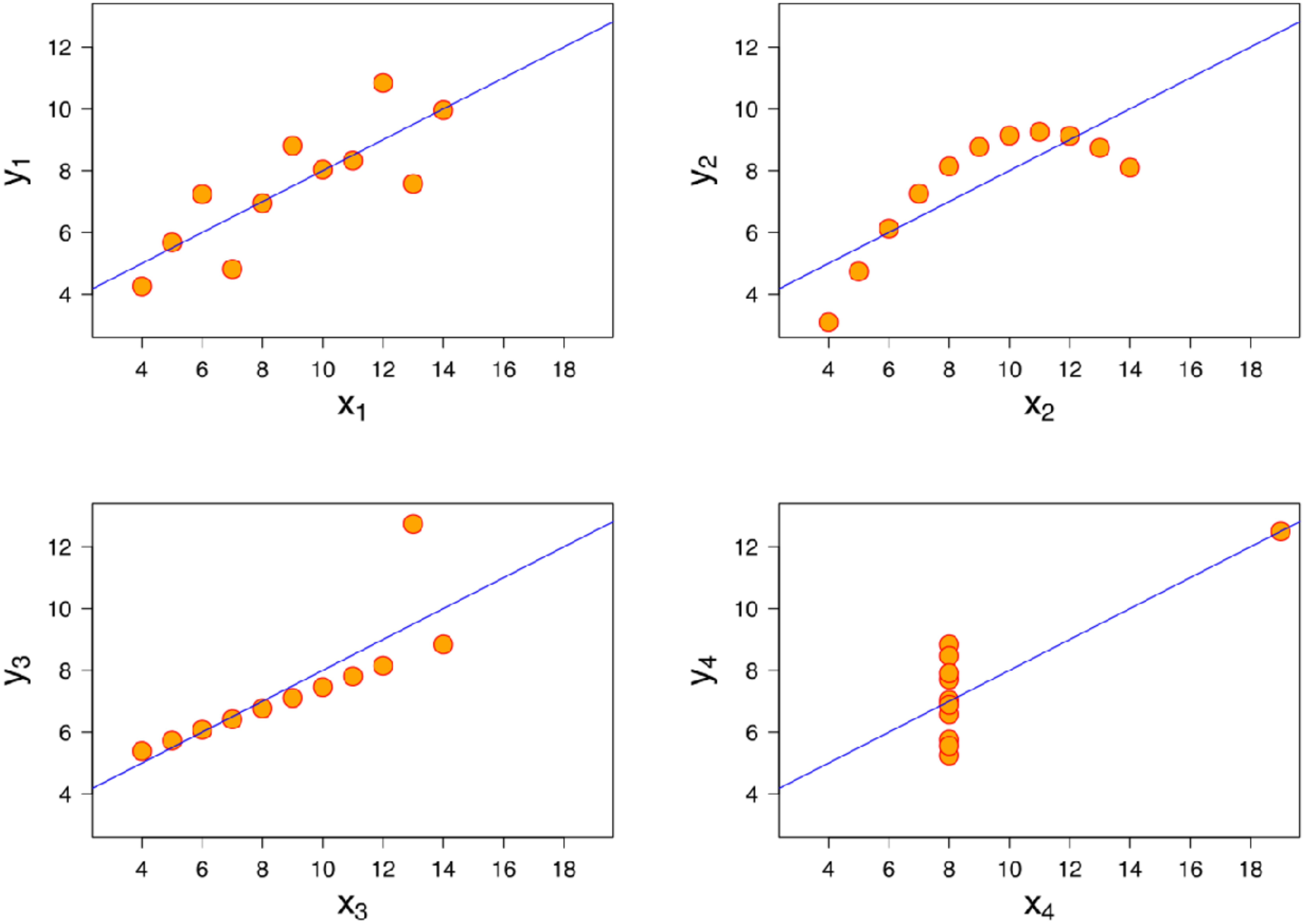
Maximum Likelihood Estimation (MLE)

- MLE of weights can be found by minimizing the Residual Sum of Squares (RSS):

$$RSS(\mathbf{w}) = \sum_{i}^{n} (y_i - \hat{y}_i)^2 = \|\mathbf{y} - \mathbf{X}^{\top}\mathbf{w}\|^2$$

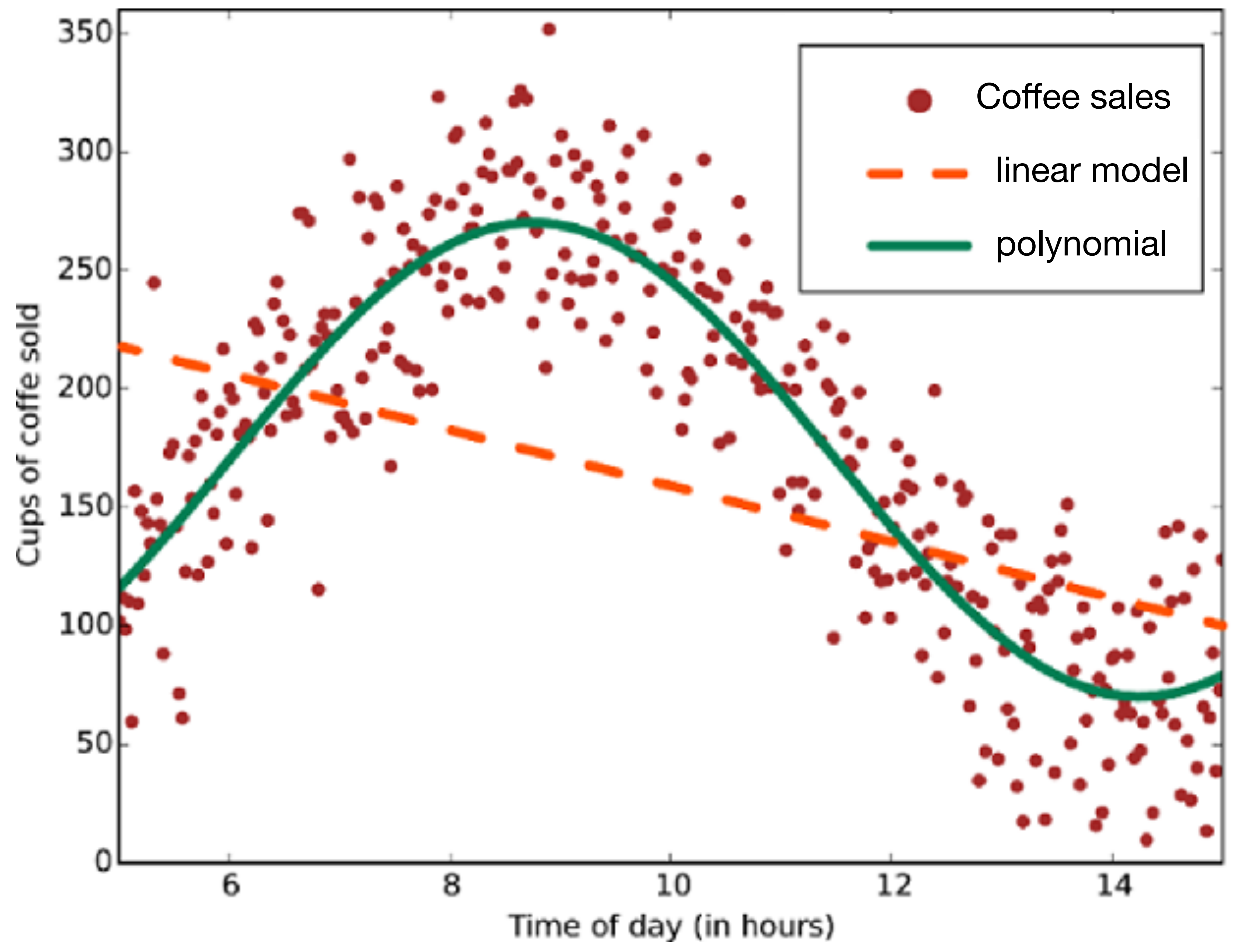
- An analytic solution is available through the Moore-Penrose psuedoinverse (Penrose, 1955): $\hat{\mathbf{w}} = \left(\mathbf{X}^{\top}\mathbf{X}\right)^{-1}\mathbf{X}^{\top}\mathbf{y}$

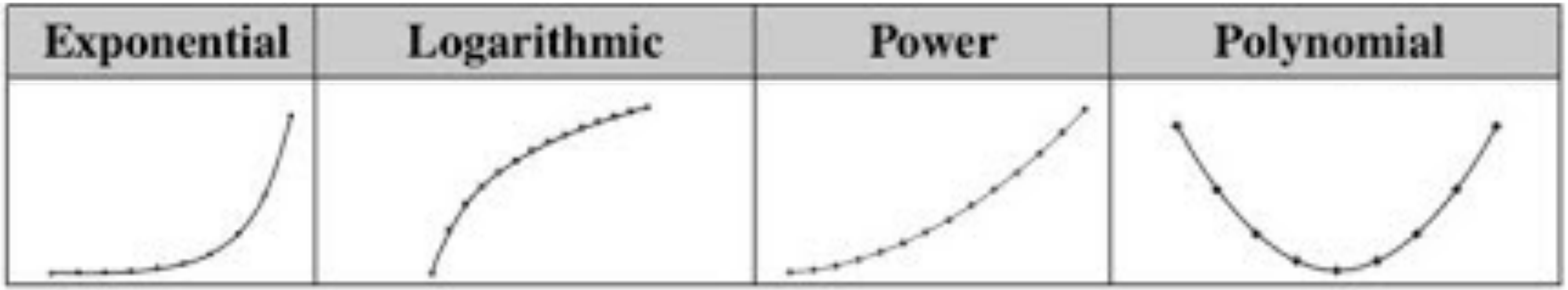
Linear assumptions don't always work



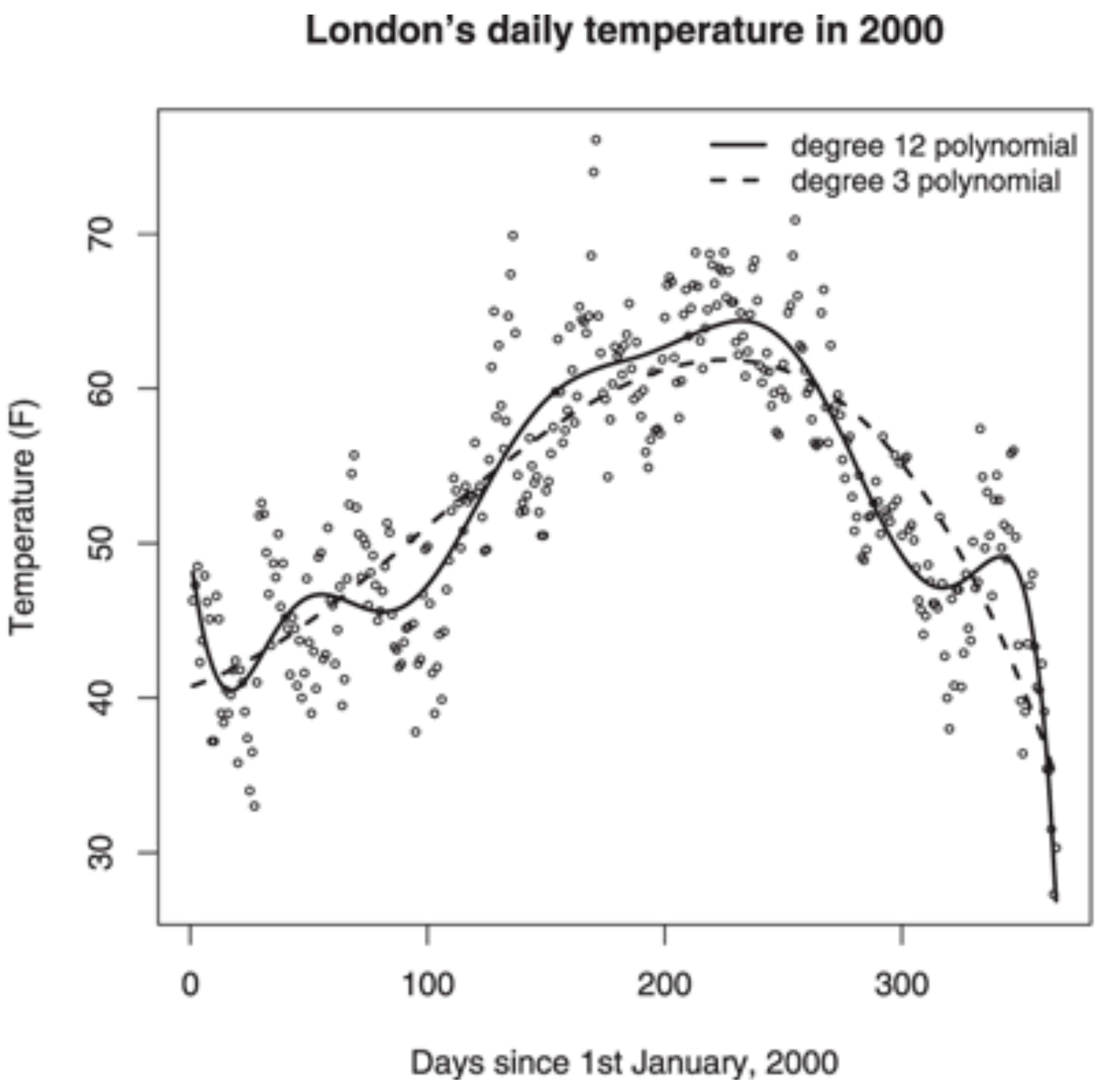
Parametric regression

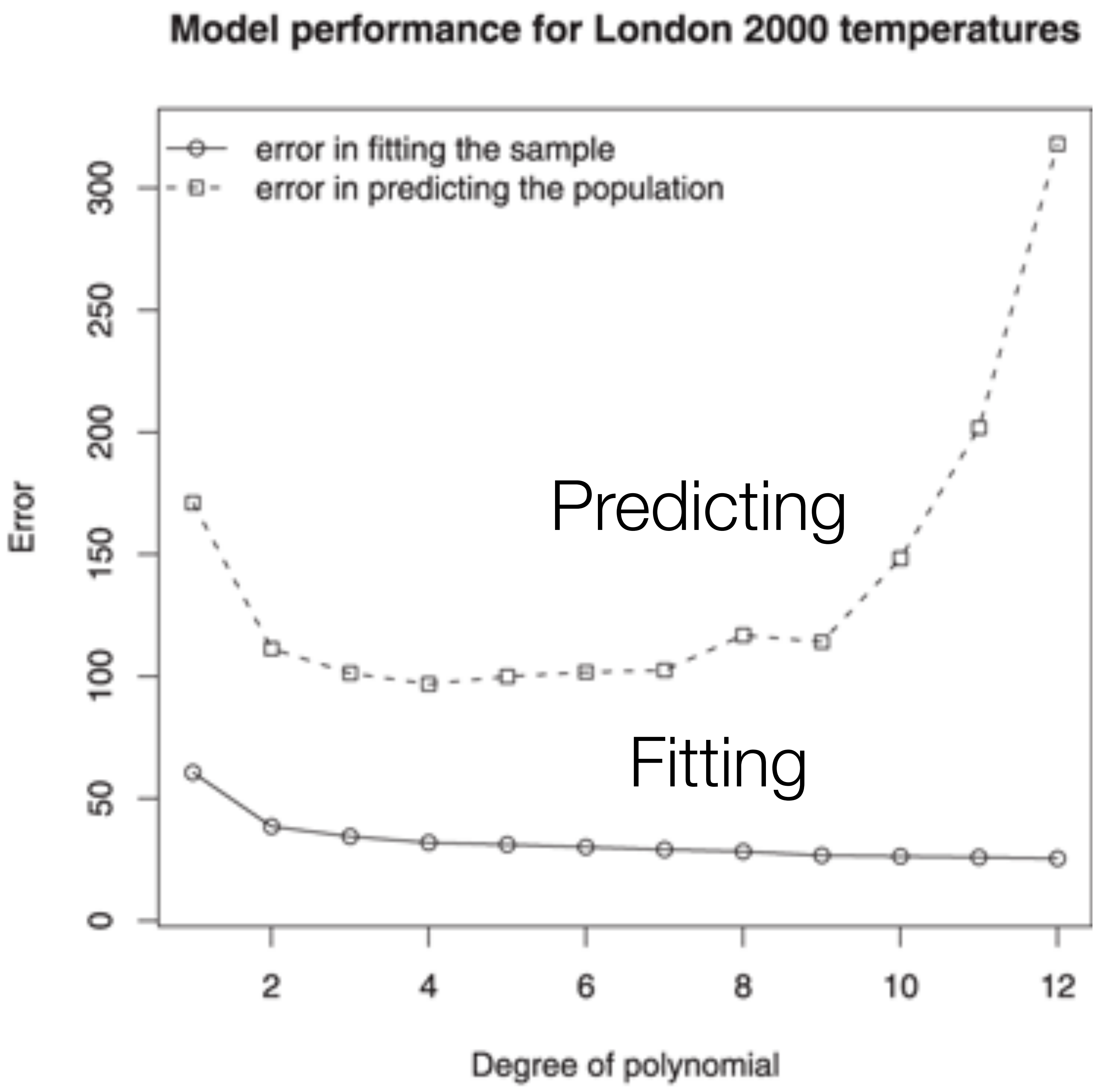
- Rather than assuming a linear relationship, assume a different functional form
  - Exponential: $f(\mathbf{x}) = \mathbf{w}^{\mathbf{x}}$
  - Logarithmic: $f(\mathbf{x}) = \mathbf{w} \log(\mathbf{x})$
  - Power: $f(\mathbf{x}) = \mathbf{x}^{\mathbf{w}}$
  - Polynomial: $f(x) = w_i x^i + w_{i-1} x^{i-1} + \ldots + w_i x$
    (switching to univariate x for simplicity)

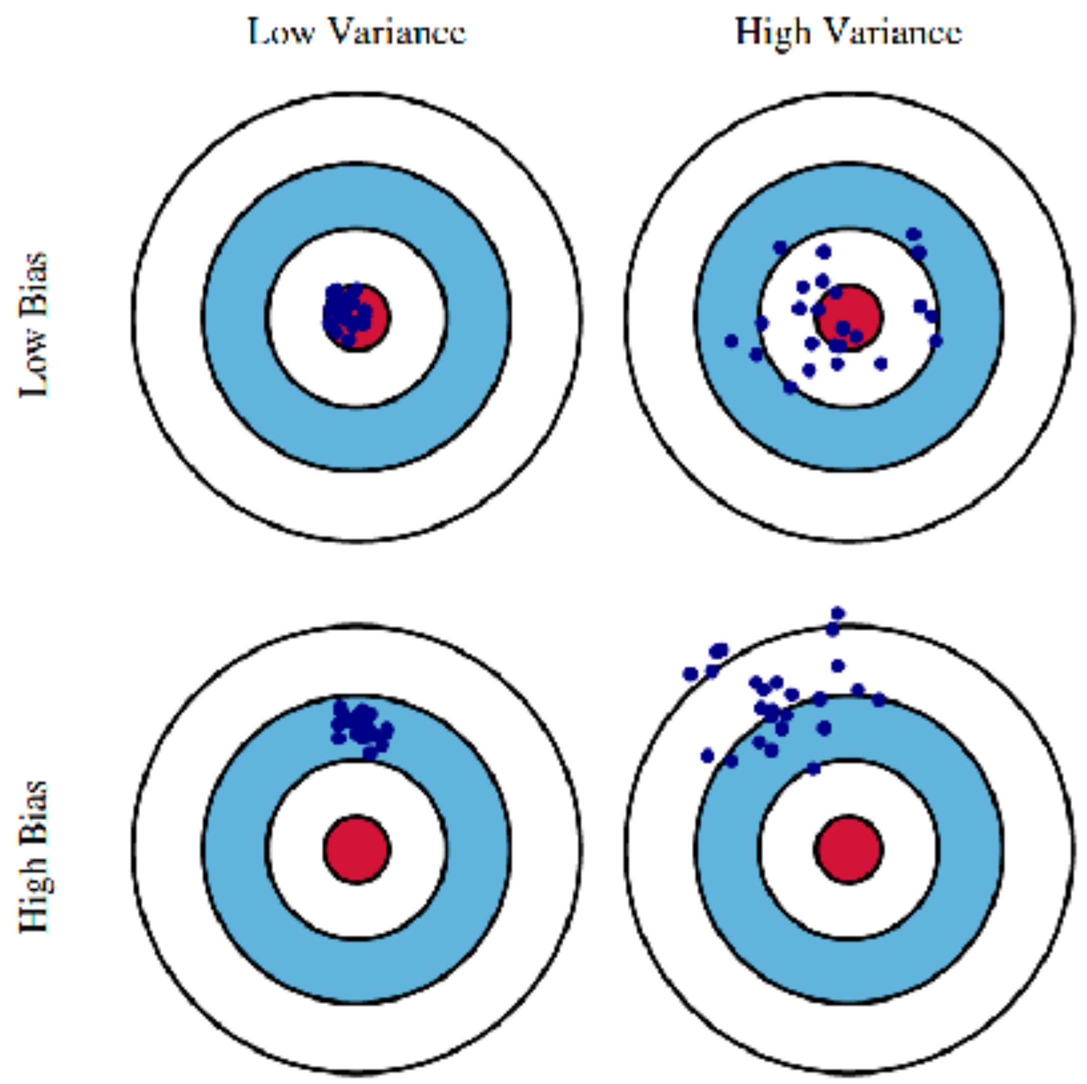


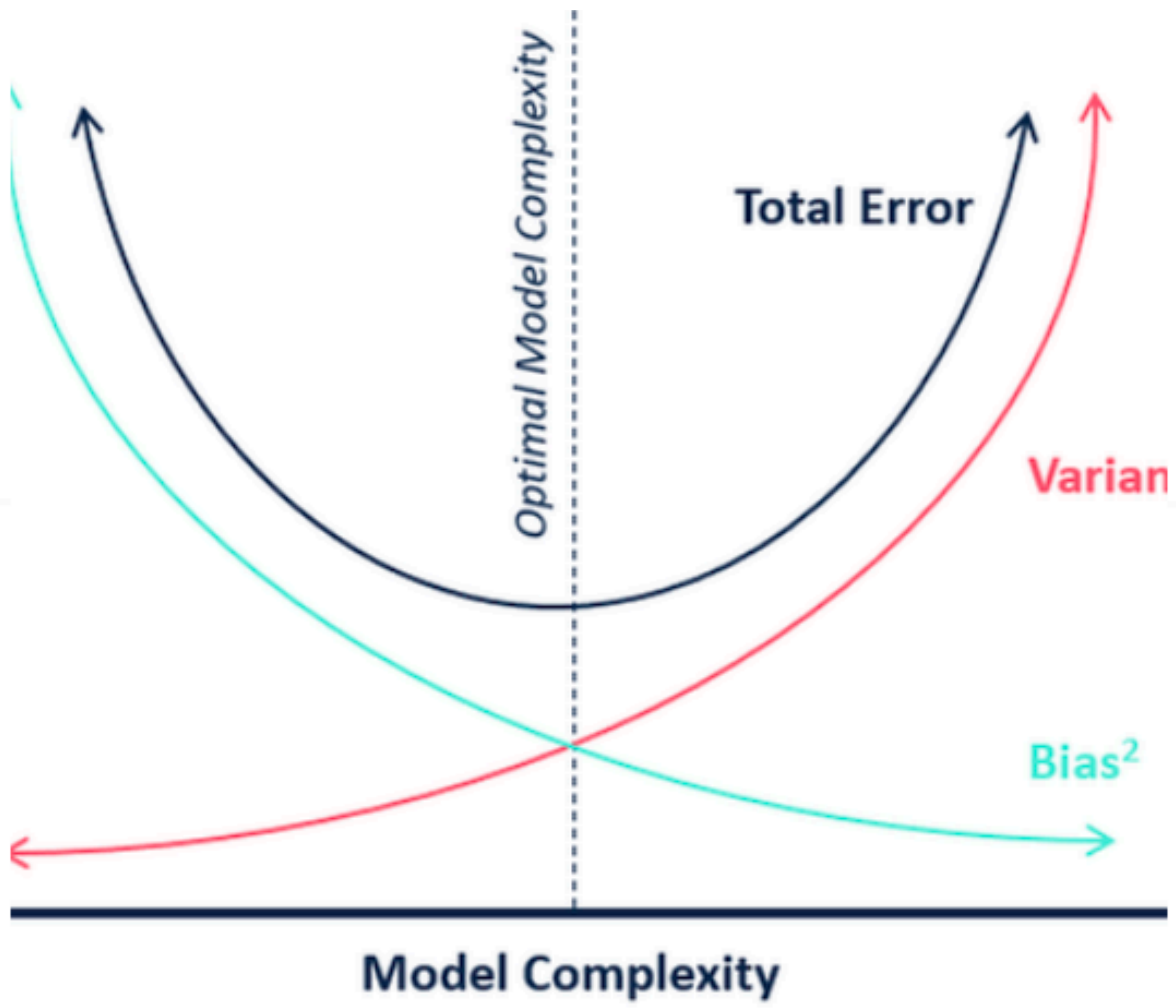


Bias-Variance trade-off





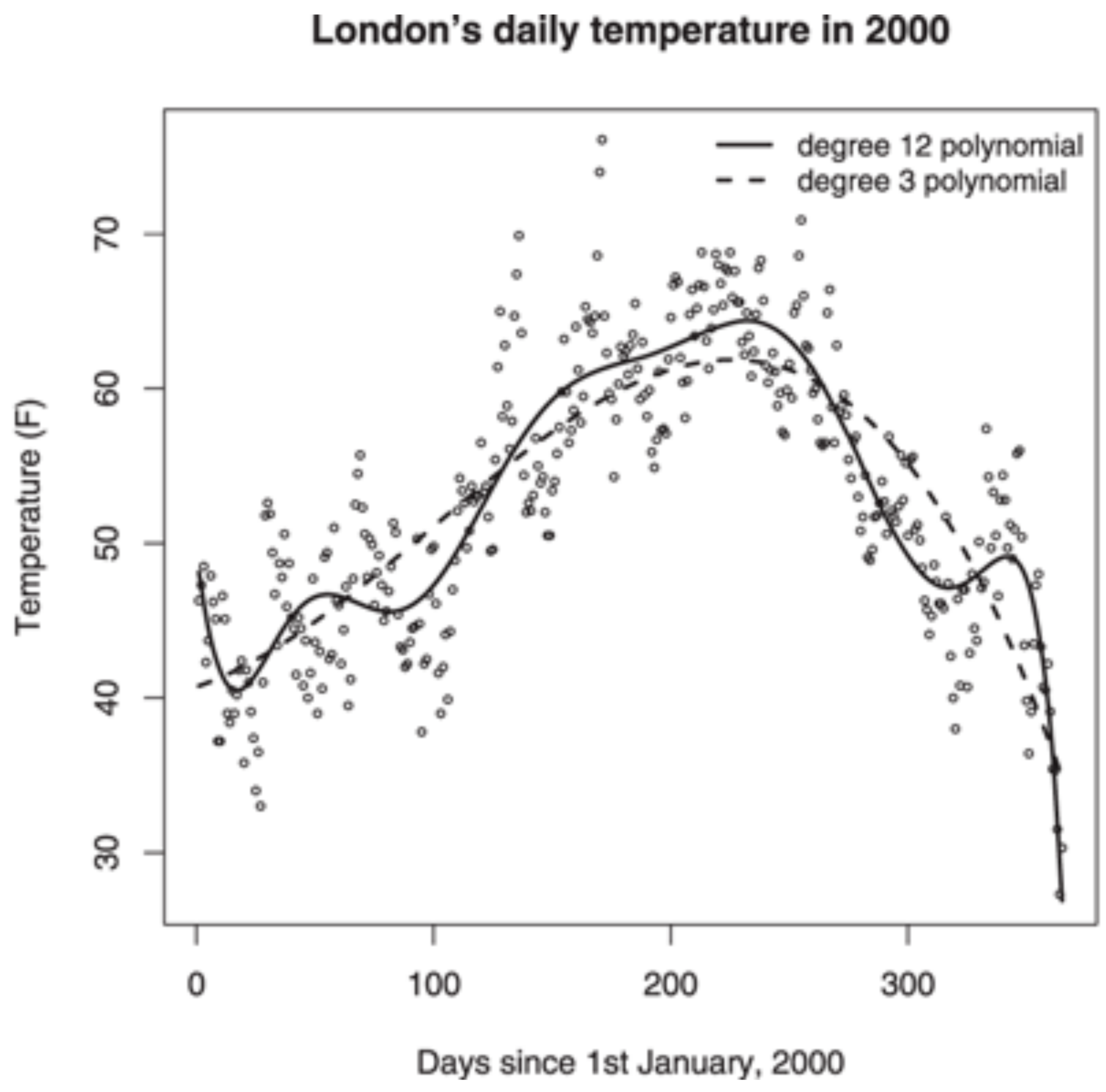


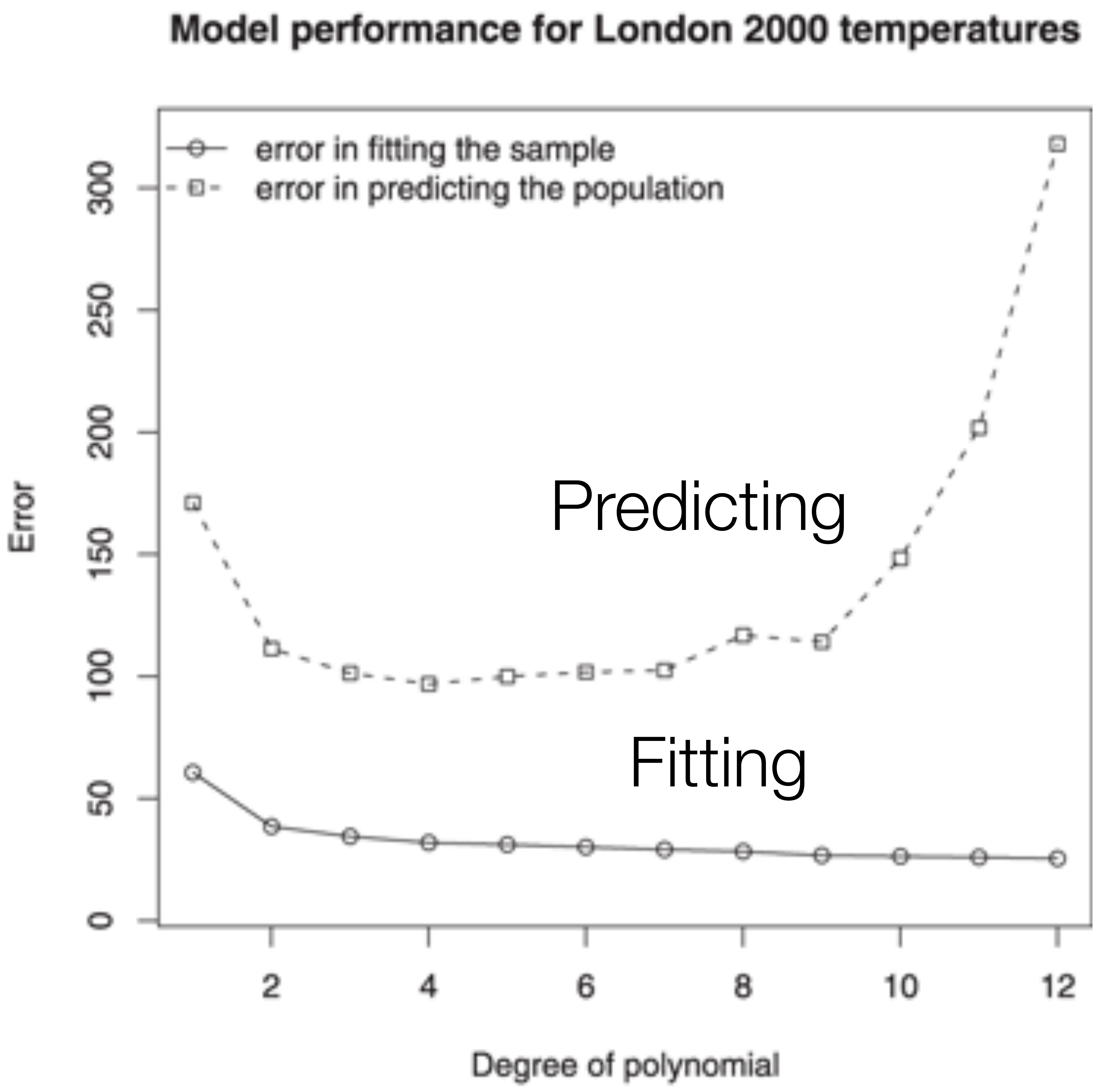


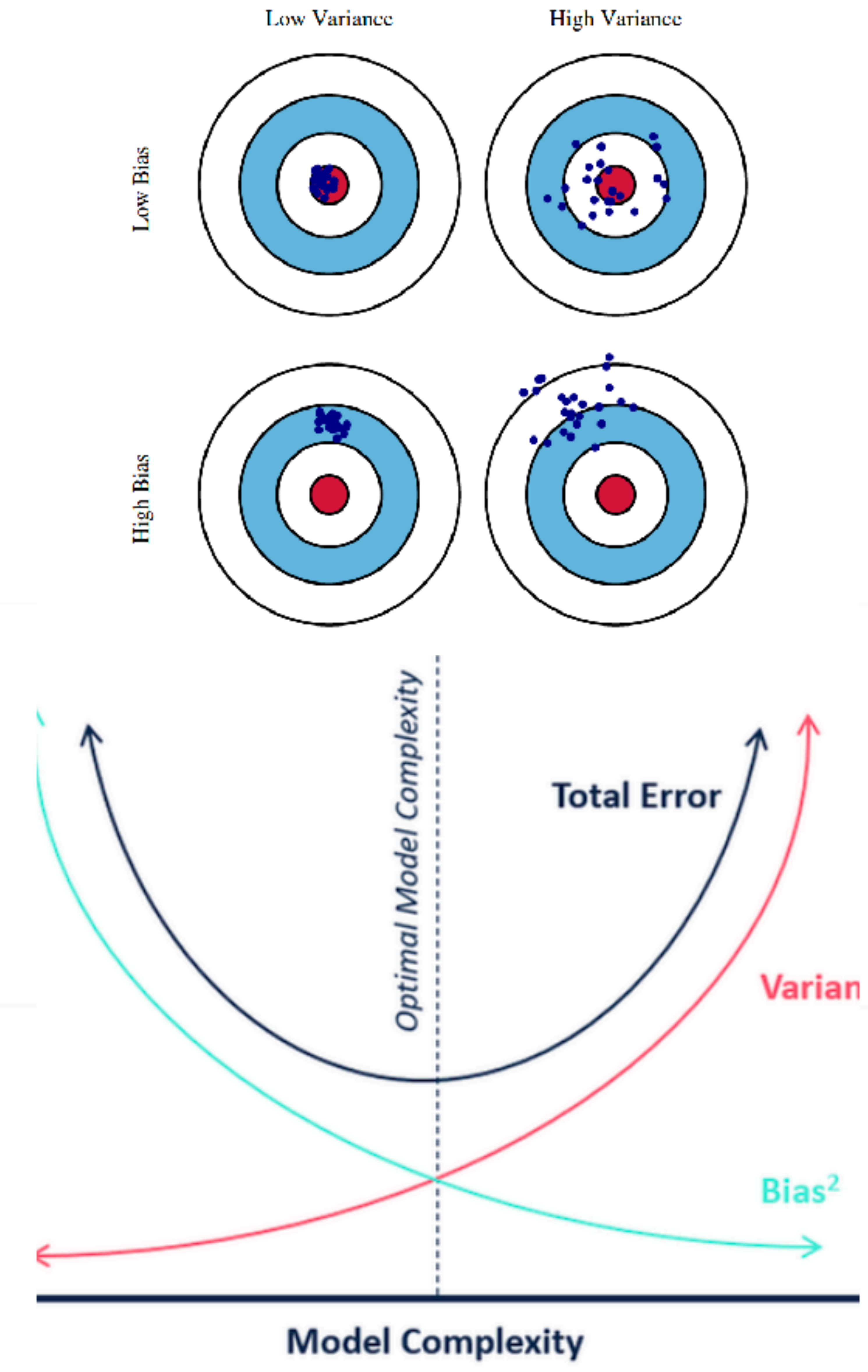
Gigerener & Brighton (*TopiCS*, 2009)

Bias-Variance trade-off

Rule-based theories don't offer guidance about how people choose between different parametric models





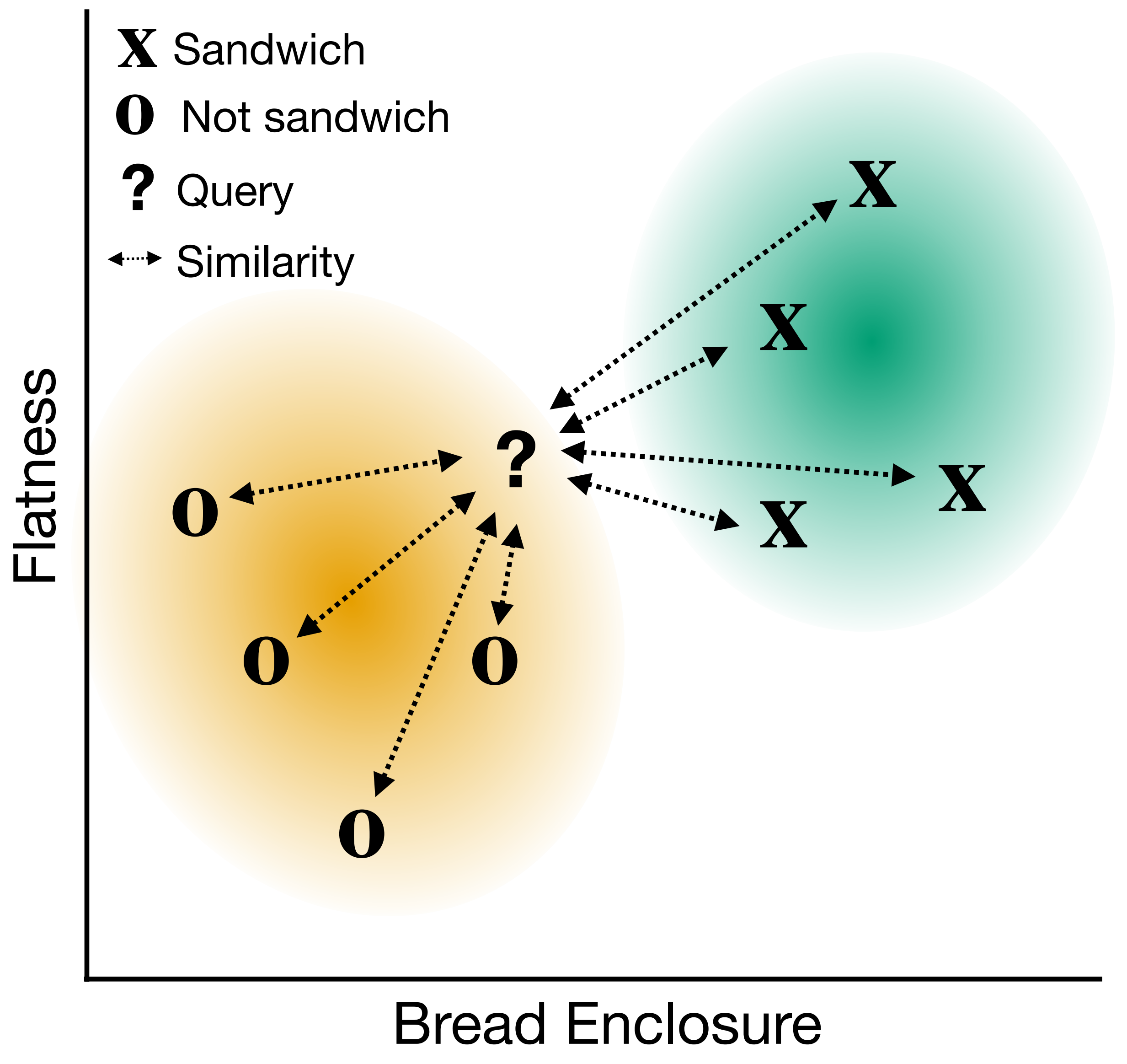


Gigerener & Brighton (*TopiCS*, 2009)

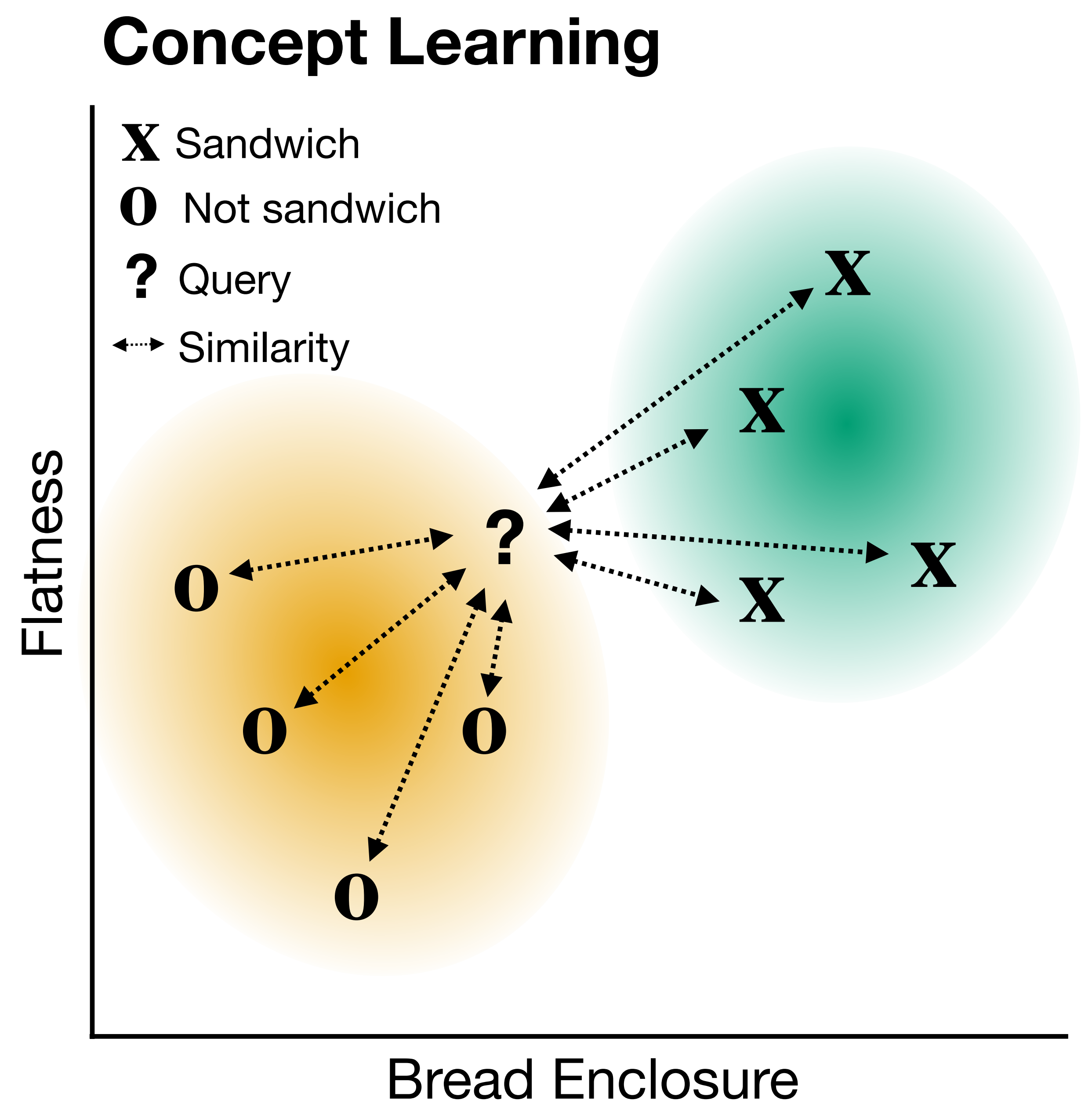
Similarity-based theories of function learning

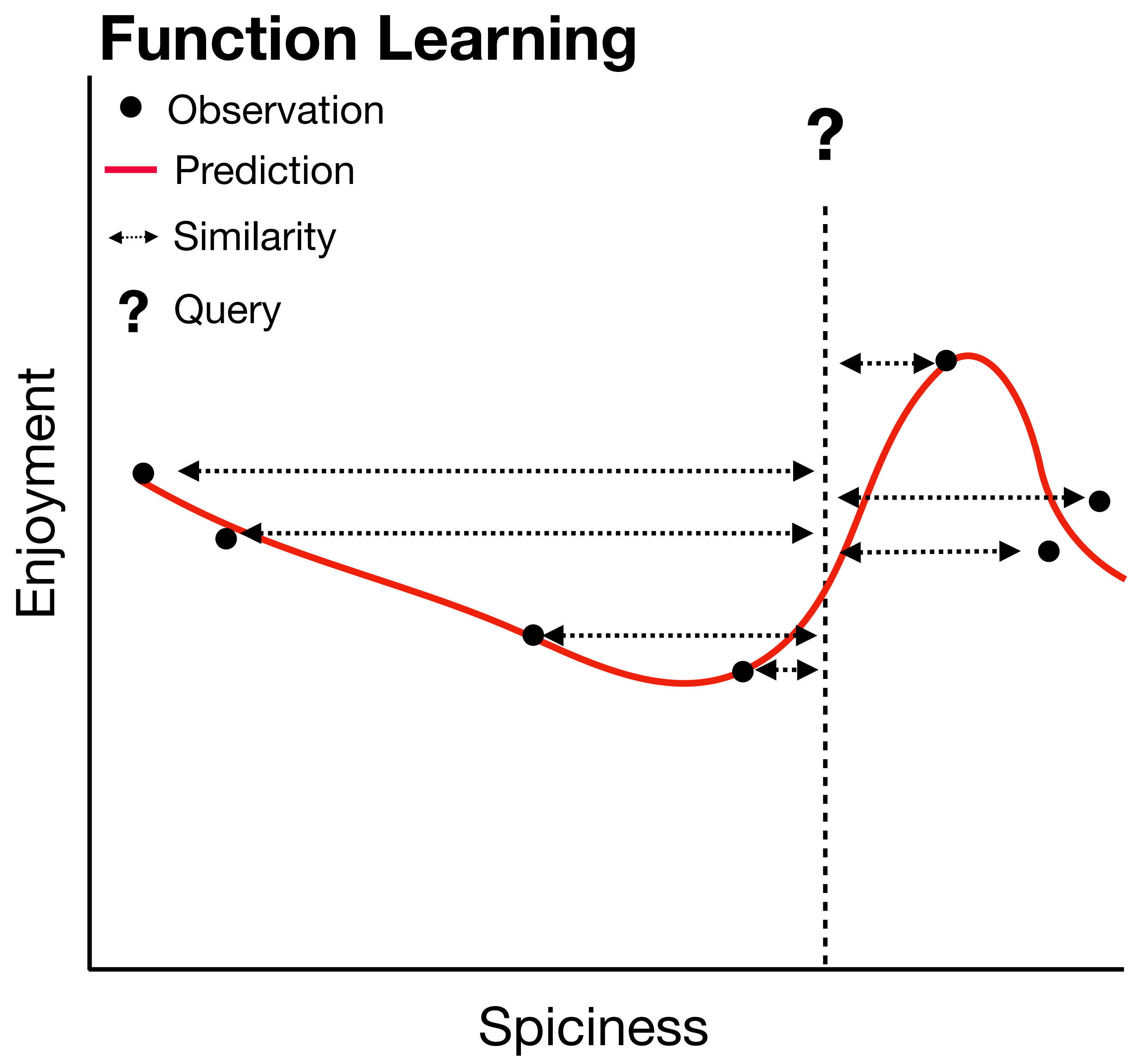
Similarity-based theories of function learning

Concept Learning



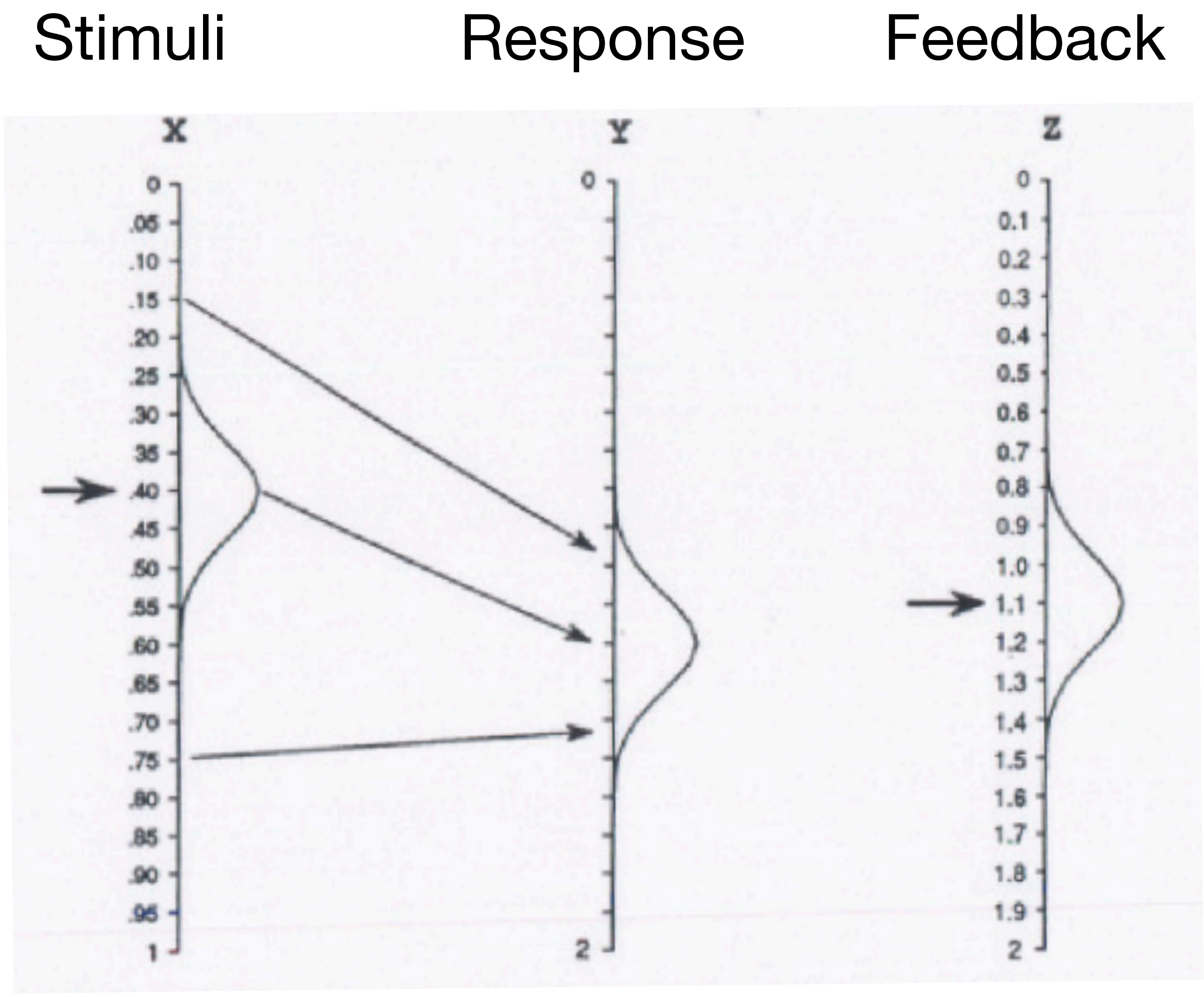
Similarity-based theories of function learning





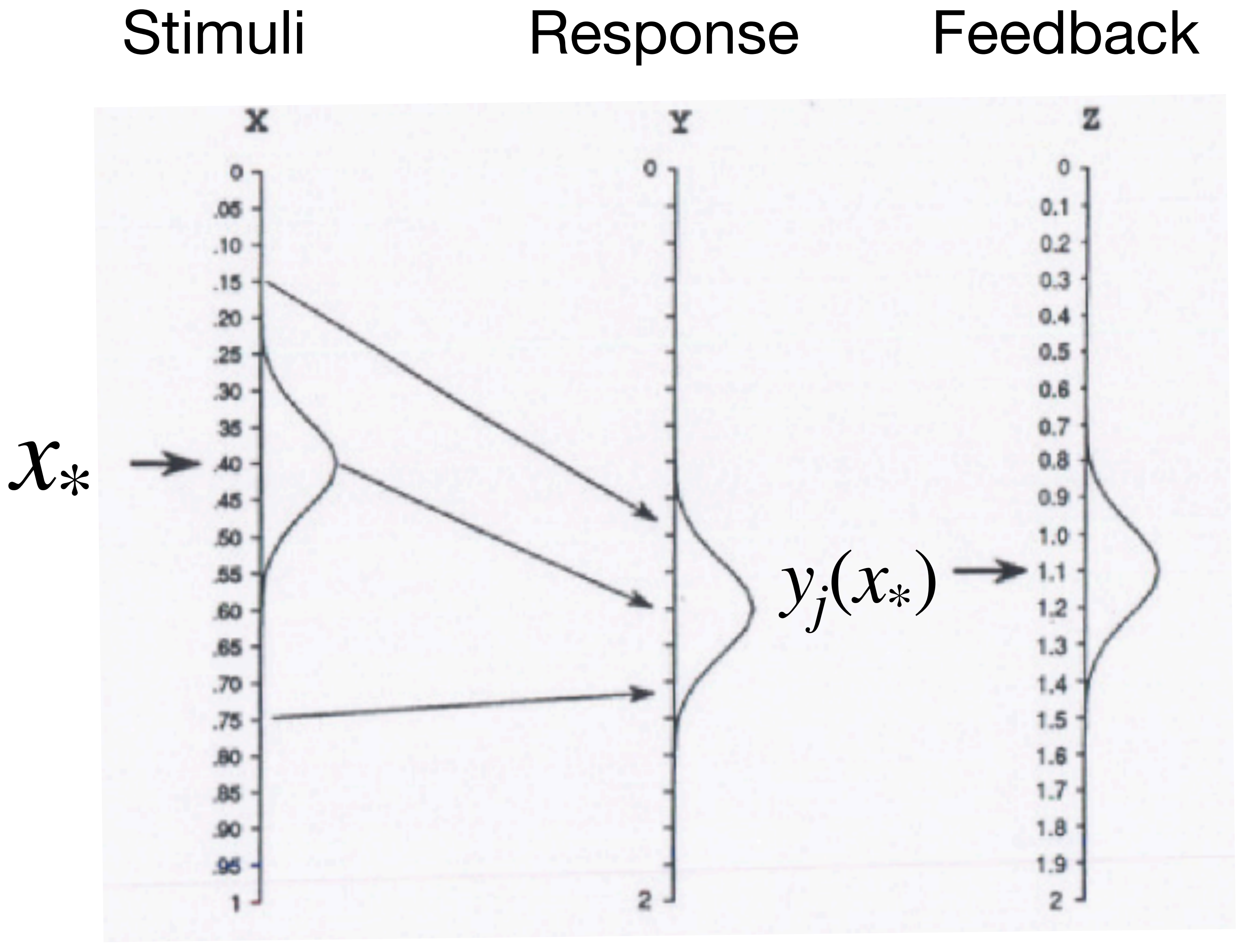
Similarity-based theories of function learning

- Associative learning model (**ALM**; Buseymeyer et al., 1997) used neural networks to encode the generic principle that *similar inputs produce similar outputs*



Similarity-based theories of function learning

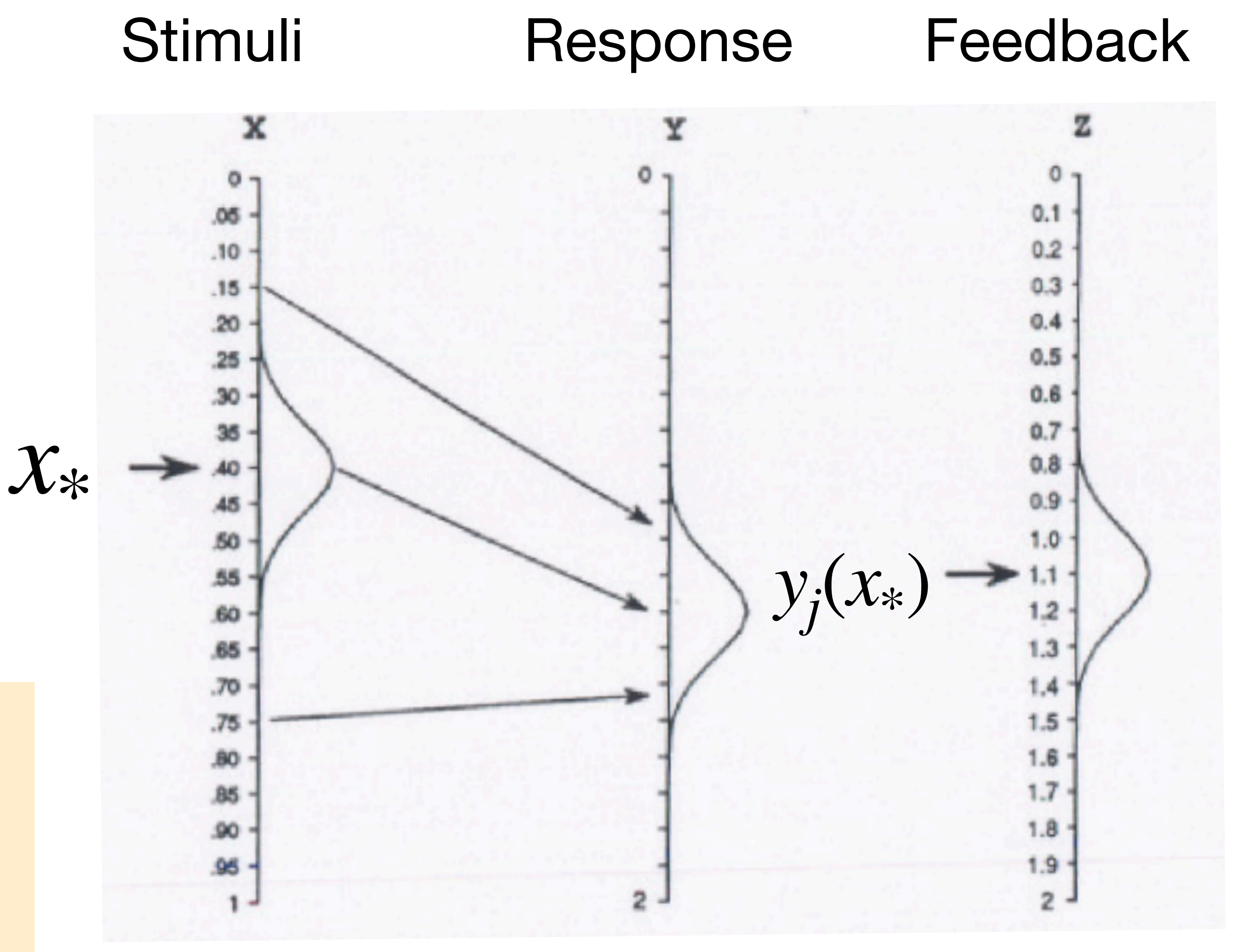
- Associative learning model (**ALM**; Buseymeyer et al., 1997) used neural networks to encode the generic principle that *similar inputs produce similar outputs*
    - TL; DR: Input x_* activates response $y_j(x_*)$ based on activation weights; weights adjusted to reduce error



Similarity-based theories of function learning

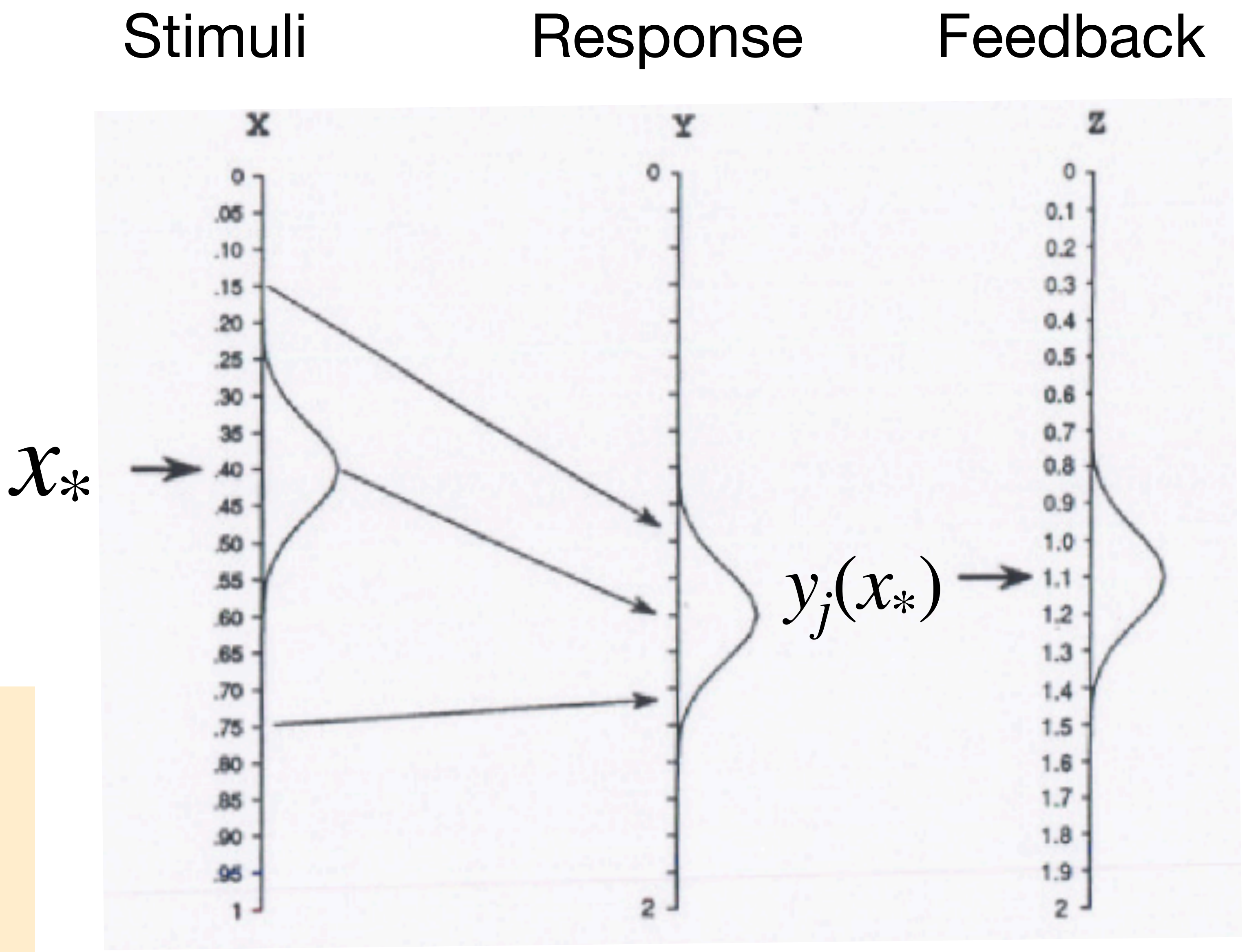
- Associative learning model (**ALM**; Buseymeyer et al., 1997) used neural networks to encode the generic principle that *similar inputs produce similar outputs*
  - TL; DR: Input x_* activates response $y_j(x_*)$ based on activation weights; weights adjusted to reduce error

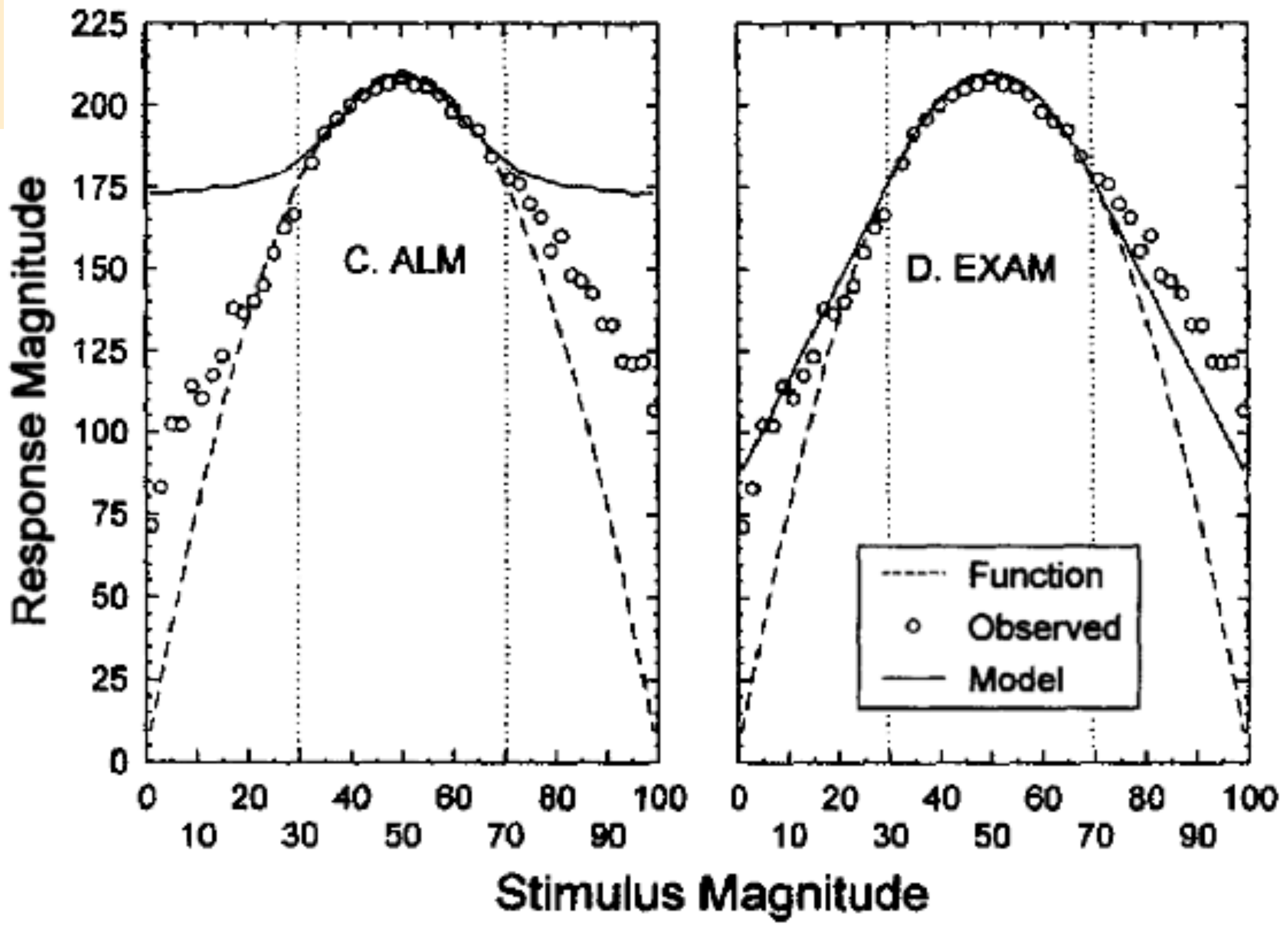
- Stimuli x_* activates input nodes according to their similarity: $a_i(x_*) = \exp\left[-\gamma(x_* - x_i)^2\right]$ where γ is a sensitivity parameter
- Output node y_j is activated according to learned weights: $y_j(x_*) = \sum_i^M w_{ji} \cdot a_i(x_*)$
- Weights updated using the delta-rule based on feedback z: $w_{ji} \leftarrow w_{ji} + \alpha\left[f_j(z) - y_j(x_*)\right] a_i(x_*)$ where $f_j(z) = \exp\left[-\gamma(z - y_j)^2\right]$



Similarity-based theories of function learning

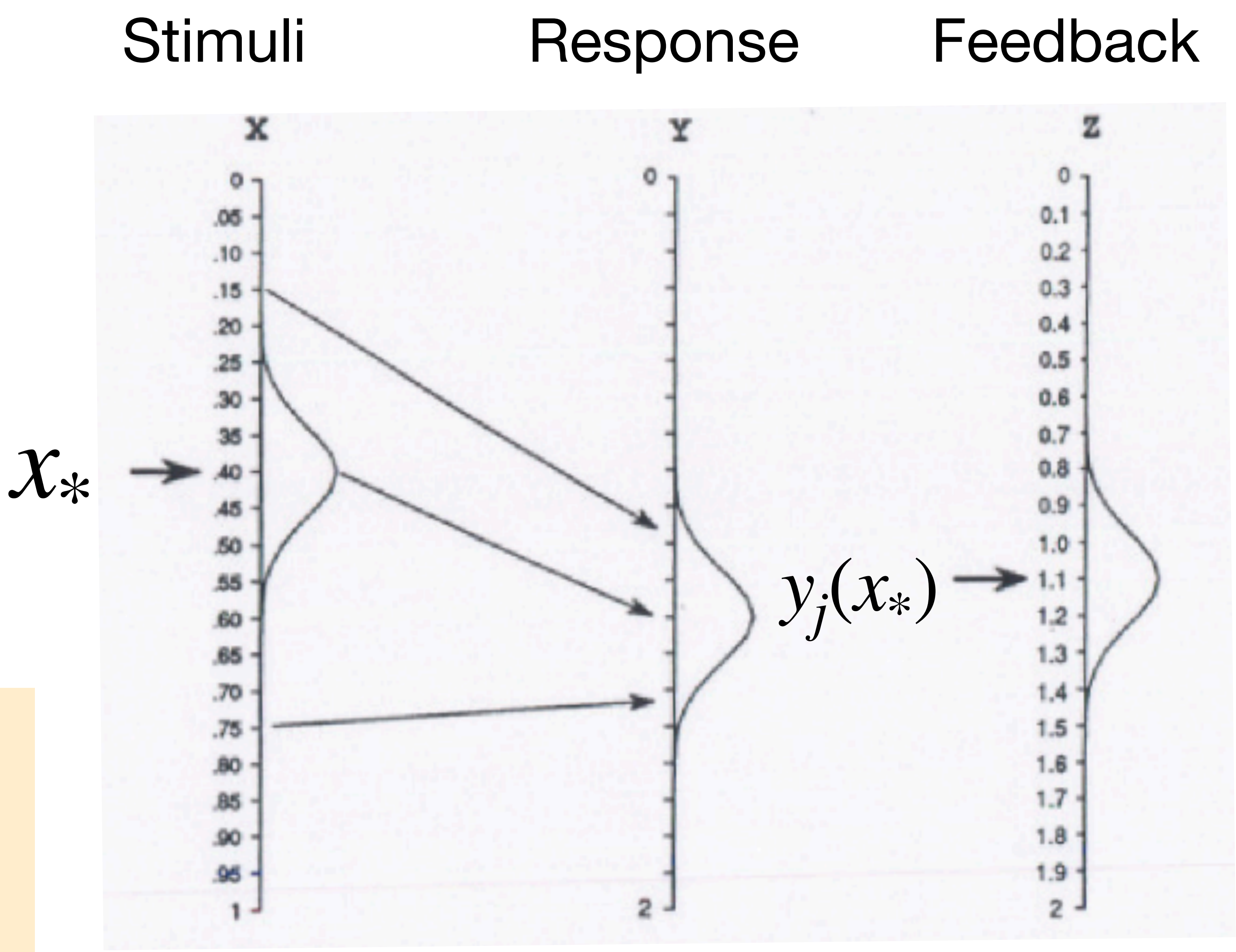
- Associative learning model (**ALM**; Buseymeyer et al., 1997) used neural networks to encode the generic principle that *similar inputs produce similar outputs*
    - TL; DR: Input x_* activates response $y_j(x_*)$ based on activation weights; weights adjusted to reduce error
        - Stimuli x_* activates input nodes according to their similarity: $a_i(x_*) = \exp\left[-\gamma(x_* - x_i)^2\right]$ where γ is a sensitivity parameter
        - Output node y_j is activated according to learned weights: $y_j(x_*) = \sum_i^M w_{ji} \cdot a_i(x_*)$
        - Weights updated using the delta-rule based on feedback z: $w_{ji} \leftarrow w_{ji} + \alpha \left[f_j(z) - y_j(x_*)\right] a_i(x_*)$ where $f_j(z) = \exp\left[-\gamma(z - y_j)^2\right]$
    - Limitation: fails to capture human extrapolation patterns

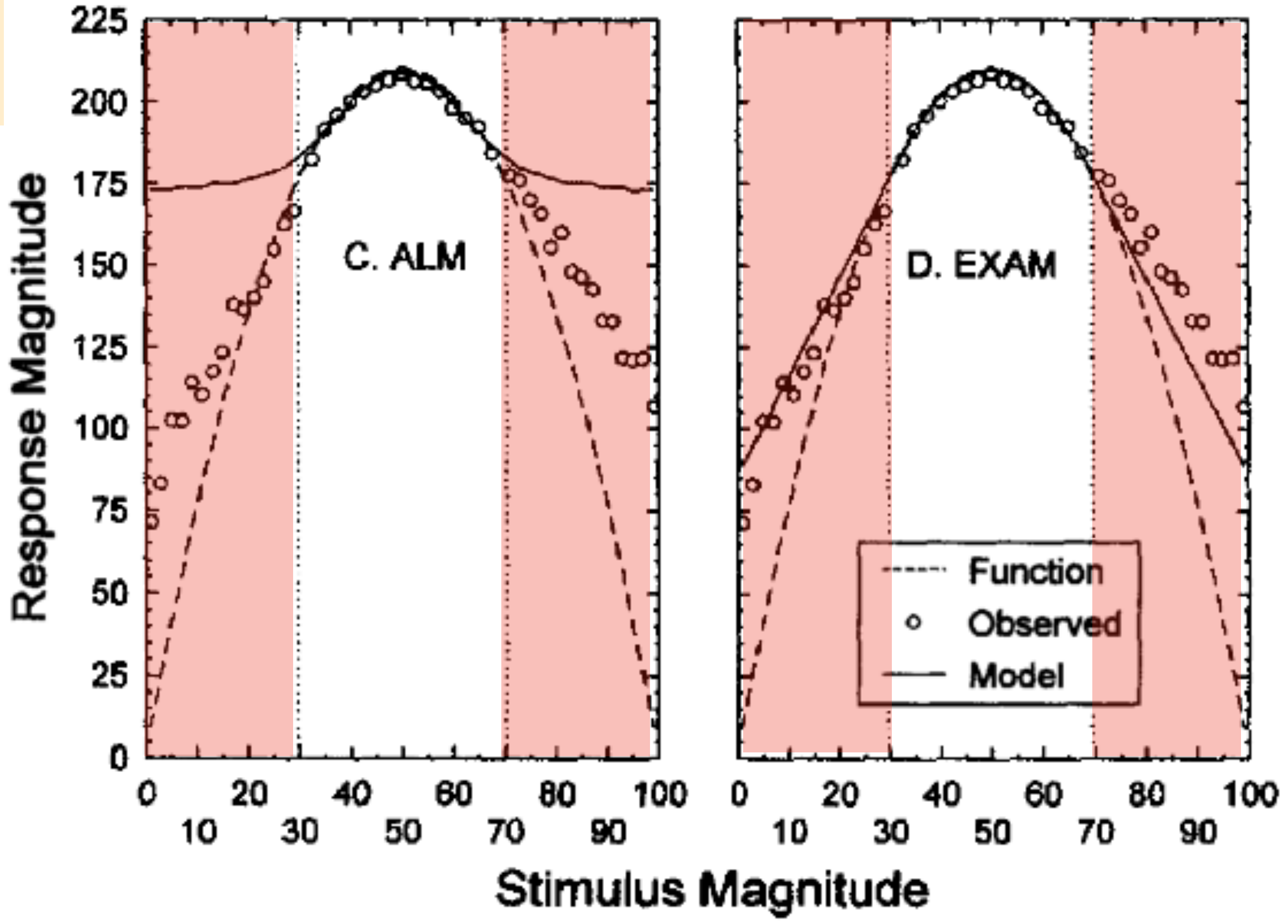




Similarity-based theories of function learning

- Associative learning model (**ALM**; Buseymeyer et al., 1997) used neural networks to encode the generic principle that *similar inputs produce similar outputs*
  - TL; DR: Input x_* activates response $y_j(x_*)$ based on activation weights; weights adjusted to reduce error
    - Stimuli x_* activates input nodes according to their similarity: $a_i(x_*) = \exp\left[-\gamma(x_* - x_i)^2\right]$ where γ is a sensitivity parameter
    - Output node y_j is activated according to learned weights: $y_j(x_*) = \sum_i^M w_{ji} \cdot a_i(x_*)$
    - Weights updated using the delta-rule based on feedback z: $w_{ji} \leftarrow w_{ji} + \alpha \left[f_j(z) - y_j(x_*)\right] a_i(x_*)$ where $f_j(z) = \exp\left[-\gamma(z - y_j)^2\right]$
  - Limitation: fails to capture human extrapolation patterns



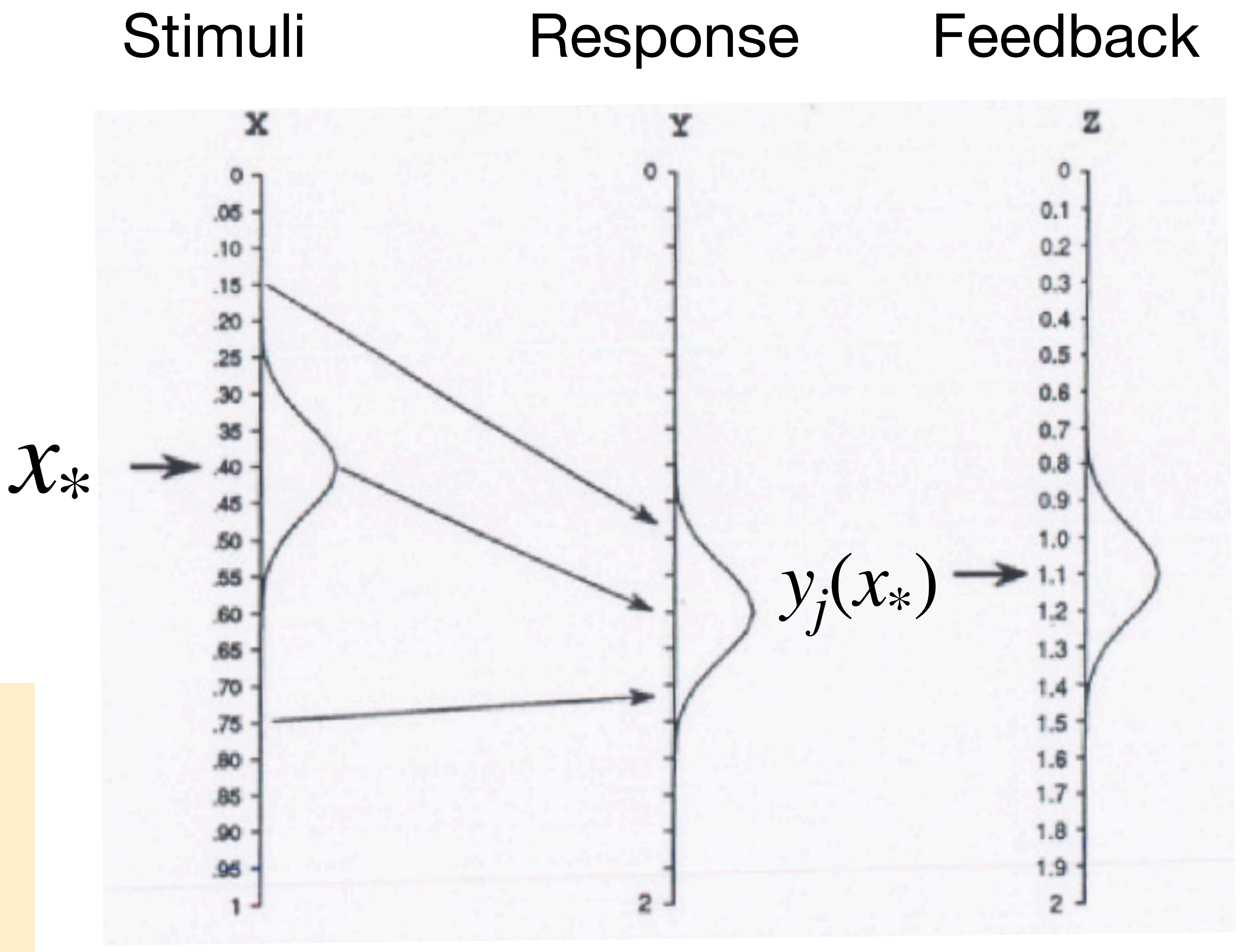


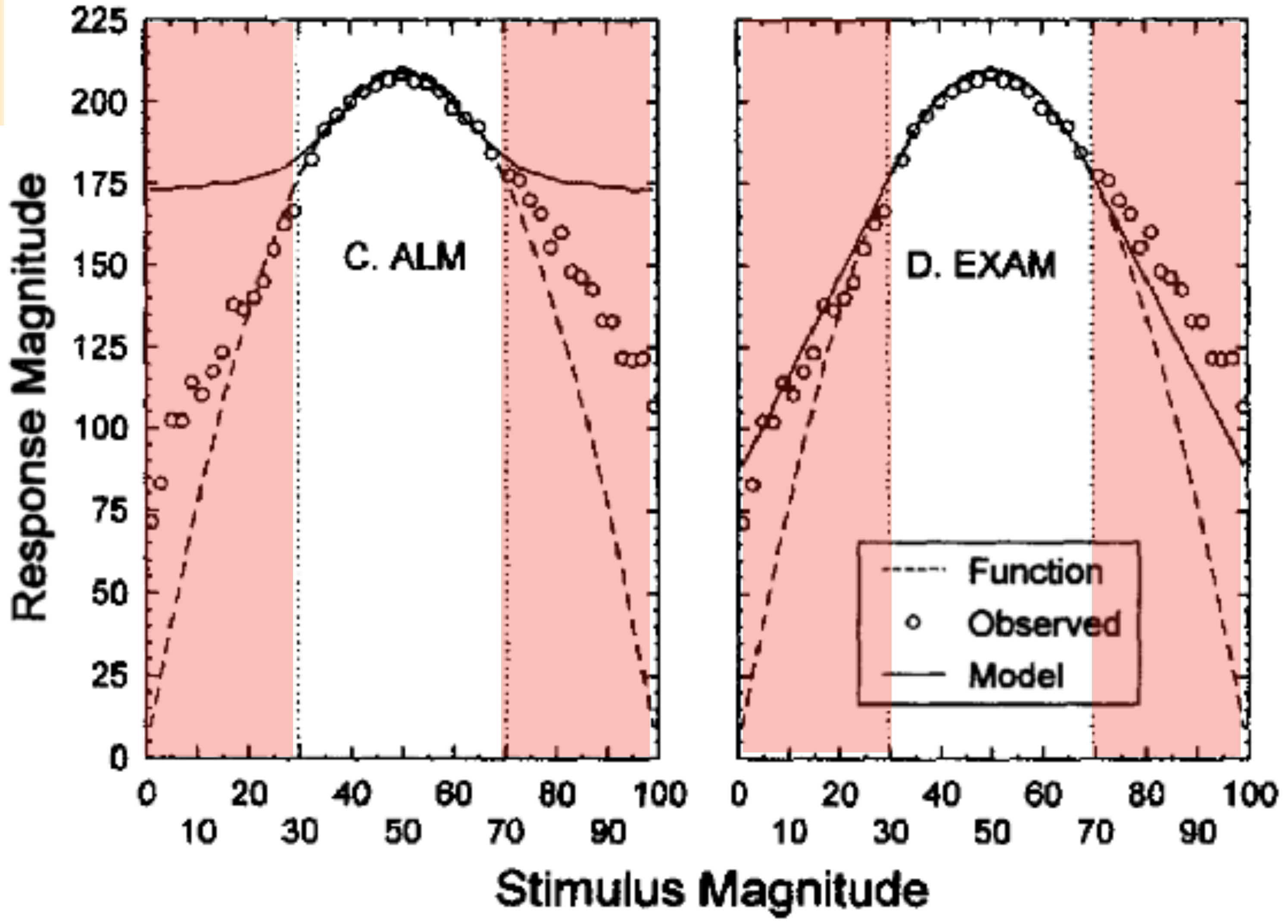
Similarity-based theories of function learning

- Associative learning model (**ALM**; Buseymeyer et al., 1997) used neural networks to encode the generic principle that *similar inputs produce similar outputs*
  - TL; DR: Input x_* activates response $y_j(x_*)$ based on activation weights; weights adjusted to reduce error

    - Stimuli x_* activates input nodes according to their similarity: $a_i(x_*) = \exp\left[-\gamma(x_* - x_i)^2\right]$ where γ is a sensitivity parameter
    - Output node y_j is activated according to learned weights: $y_j(x_*) = \sum_i^M w_{ji} \cdot a_i(x_*)$
    - Weights updated using the delta-rule based on feedback z: $w_{ji} \leftarrow w_{ji} + \alpha\left[f_j(z) - y_j(x_*)\right] a_i(x_*)$ where $f_j(z) = \exp\left[-\gamma(z - y_j)^2\right]$

  - Limitation: fails to capture human extrapolation patterns
- Extrapolation-Association Model (**EXAM**; Delosh et al,. 1997) extends ALM by adding a linear approximation of ALM outputs to account for more linear extrapolation patterns in humans



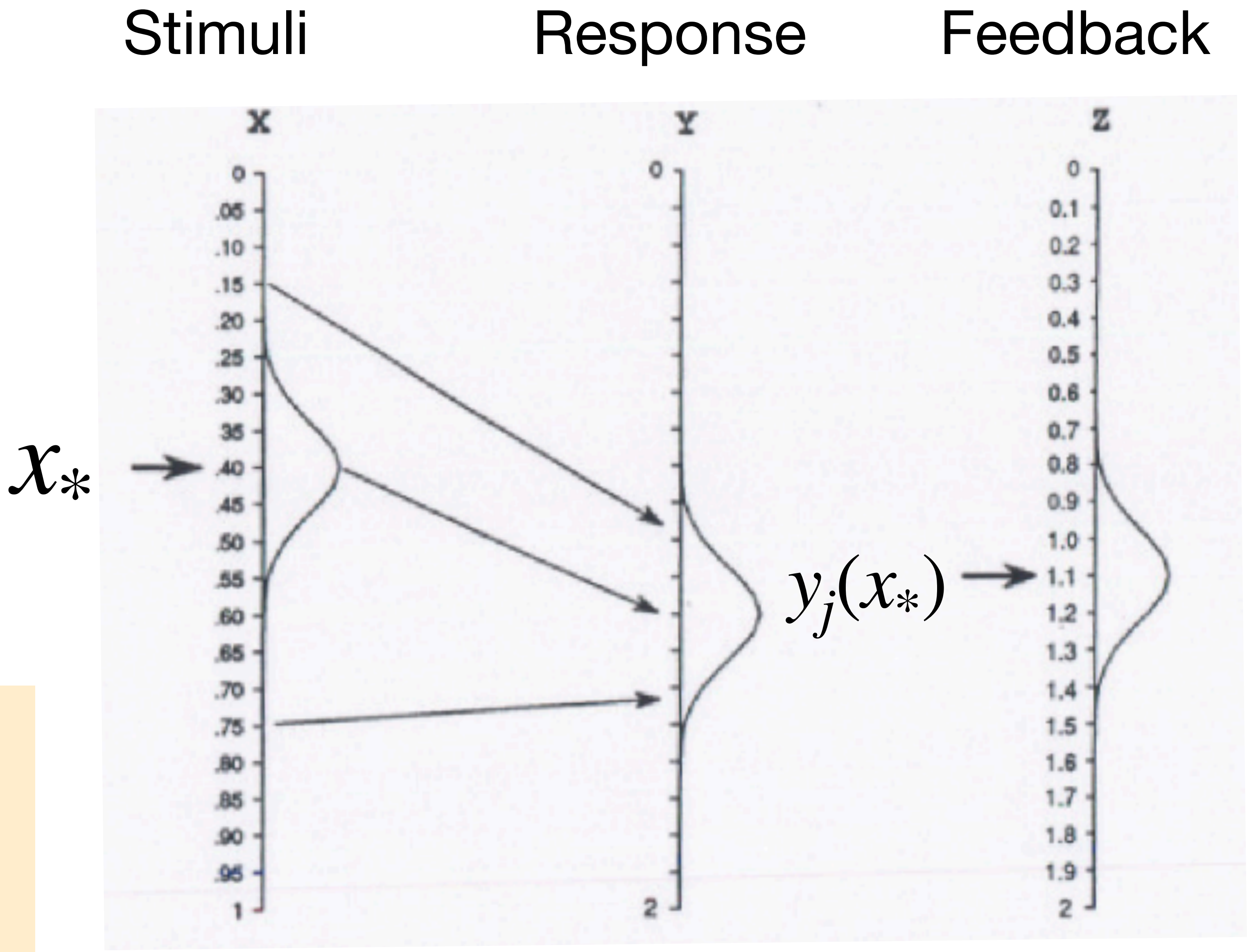


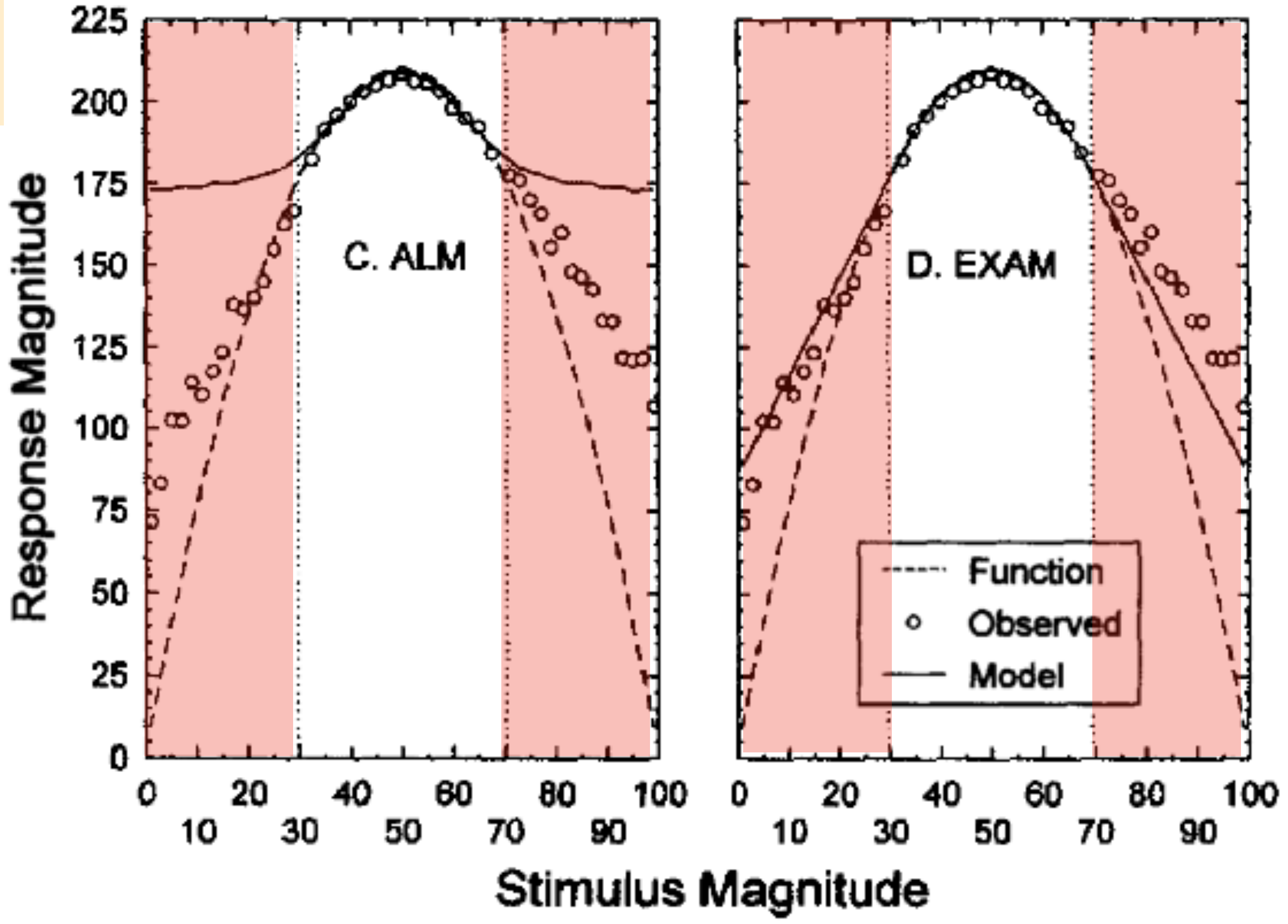
Similarity-based theories of function learning

- Associative learning model (**ALM**; Buseymeyer et al., 1997) used neural networks to encode the generic principle that *similar inputs produce similar outputs*
    - TL; DR: Input x_* activates response $y_j(x_*)$ based on activation weights; weights adjusted to reduce error

- Stimuli x_* activates input nodes according to their similarity: $a_i(x_*) = \exp\left[-\gamma(x_* - x_i)^2\right]$ where γ is a sensitivity parameter
- Output node y_j is activated according to learned weights: $y_j(x_*) = \sum_i^M w_{ji} \cdot a_i(x_*)$
- Weights updated using the delta-rule based on feedback z: $w_{ji} \leftarrow w_{ji} + \alpha\left[f_j(z) - y_j(x_*)\right] a_i(x_*)$ where $f_j(z) = \exp\left[-\gamma(z - y_j)^2\right]$

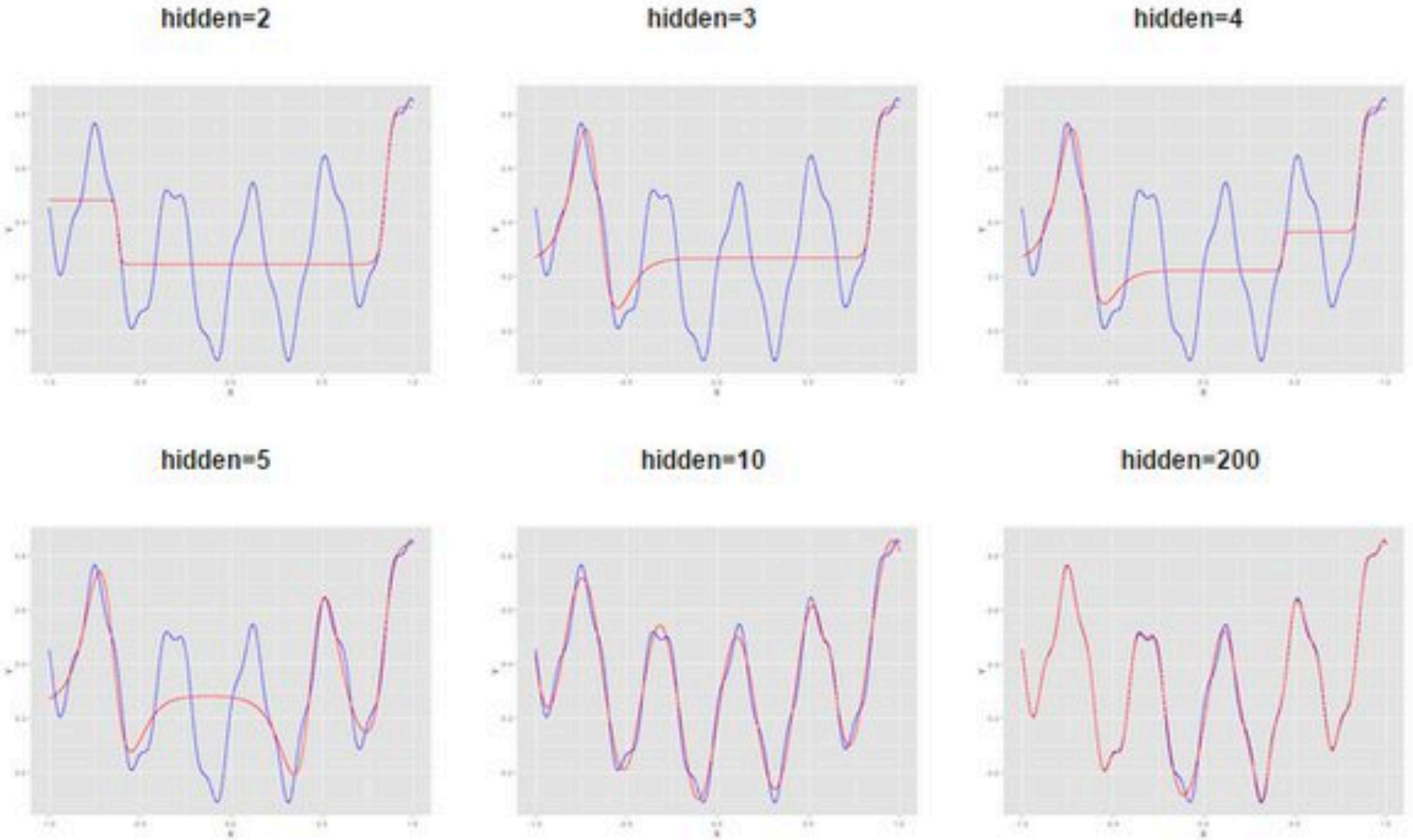
    - Limitation: fails to capture human extrapolation patterns
- Extrapolation-Association Model (**EXAM**; Delosh et al,. 1997) extends ALM by adding a linear approximation of ALM outputs to account for more linear extrapolation patterns in humans
    - But humans also sometimes extrapolate in a non-linear fashion (Bott & Heit, 2004)





Neural networks as Universal Function Approximators

- Recall Cybenko (1989): Every continuous function can be approximated arbitrarily closely by an MLP with just a single hidden layer
  - adding more nodes in the hidden layer increases the representational capacity of the network
- But fitting is not the same as predicting
- As we see from ALM, extrapolation patterns of NNs don't always match the inductive biases of humans learners

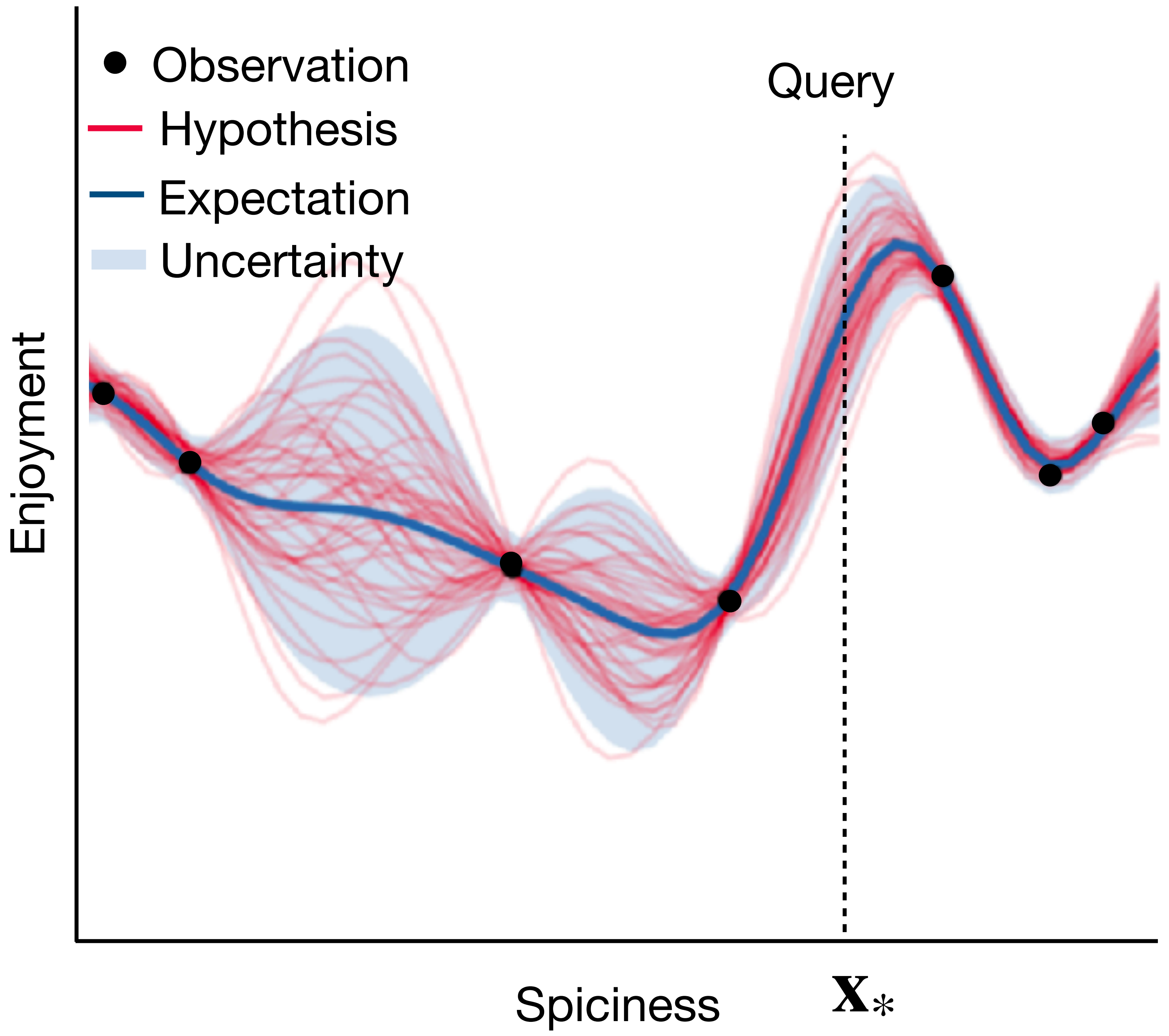


Gaussian Process (GP) regression as a hybrid model

- Bayesian framework for function learning
  - Assumes a distribution over functions: each function corresponds to a **hypothesis** about the relationship between x and y
- Bayesian posterior is conditioned on past **observations**, letting us make predictions (with uncertainty) about any point along the input space ($\mathbf{x}_*$)
- Called Gaussian process, because of Gaussian assumptions: predictions at each point are defined by a posterior **mean** (i.e., expectation) and **variance** (uncertainty); more details on the next slide
- GPs are a *non-parametric* model, meaning the complexity is defined by the data not the number of parameters in the chosen functional class (i.e., *parametric models*)

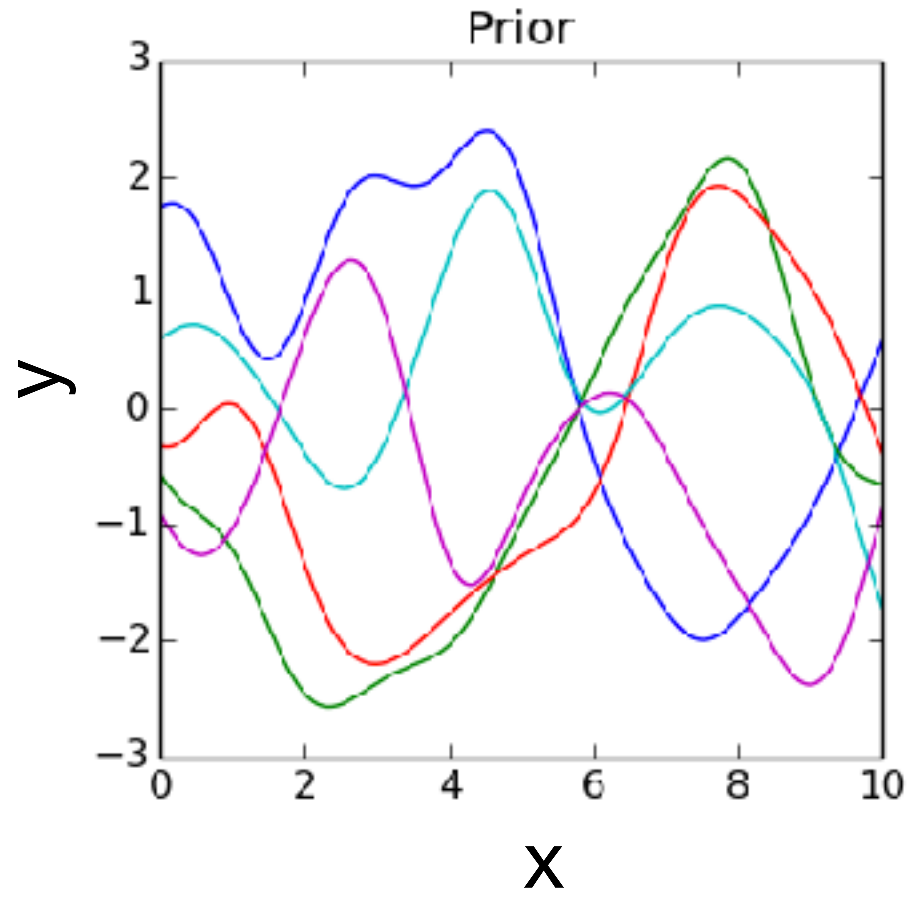
Gaussian Process (GP) regression as a hybrid model

- Bayesian framework for function learning
  - Assumes a distribution over functions: each function corresponds to a **hypothesis** about the relationship between x and y
- Bayesian posterior is conditioned on past **observations**, letting us make predictions (with uncertainty) about any point along the input space ($\mathbf{x}_*$)
- Called Gaussian process, because of Gaussian assumptions: predictions at each point are defined by a posterior **mean** (i.e., expectation) and **variance** (uncertainty); more details on the next slide
- GPs are a *non-parametric* model, meaning the complexity is defined by the data not the number of parameters in the chosen functional class (i.e., *parametric models*)



see Rasmussen & Williams (2005) for further details

Gaussian Process (GP) regression in detail

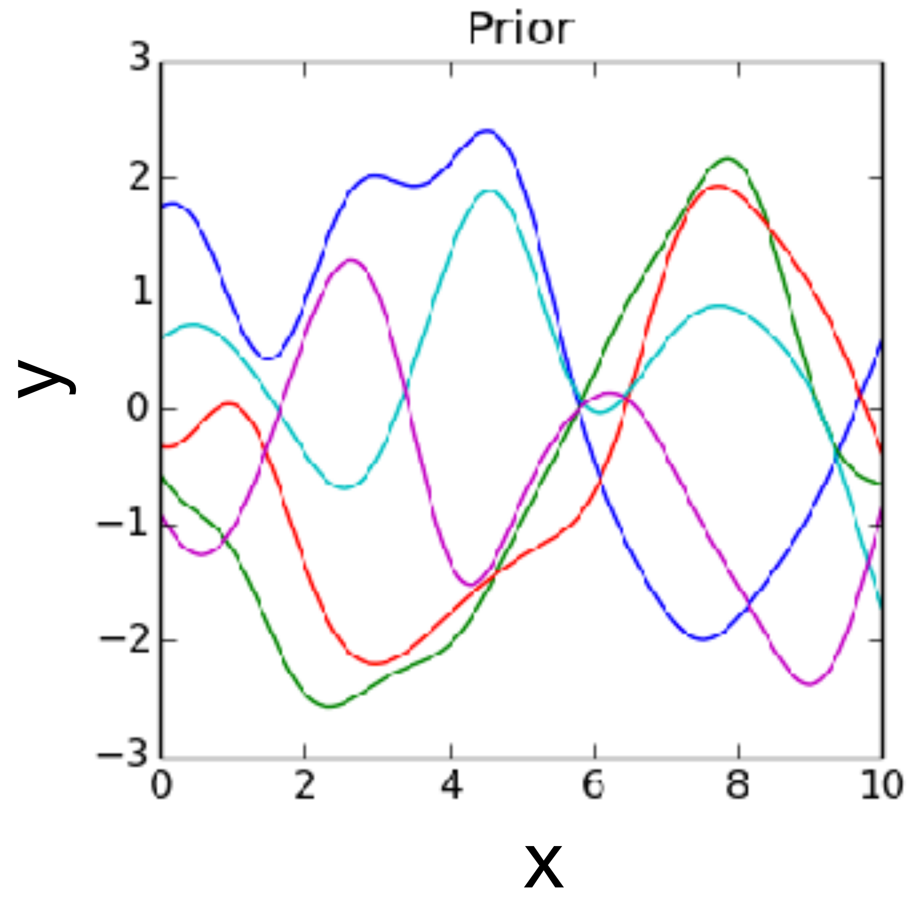


- Prior over functions (i.e., hypotheses) is a multivariate Gaussian:
  $P(f) \sim \mathcal{GP}\left(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')\right)$
  - prior mean $m(\mathbf{x})$ is typically set to 0 without loss of generalization
  - Covariance $k(\mathbf{x}, \mathbf{x}')$ is defined by a choice of kernel e.g., RBF kernel:
    $$k(\mathbf{x}, \mathbf{x}') = \exp\left(\frac{-||\mathbf{x} - \mathbf{x}'||^2}{2\lambda^2}\right)$$
    where λ defines the expected smoothness of the function
- Once we acqire some data $\mathcal{D} = \{\mathbf{X}, \mathbf{y}\}$, we can compute a posterior prediction about any new datapoint $\mathbf{x}_*$ that is also Gaussian with mean and variance defined as

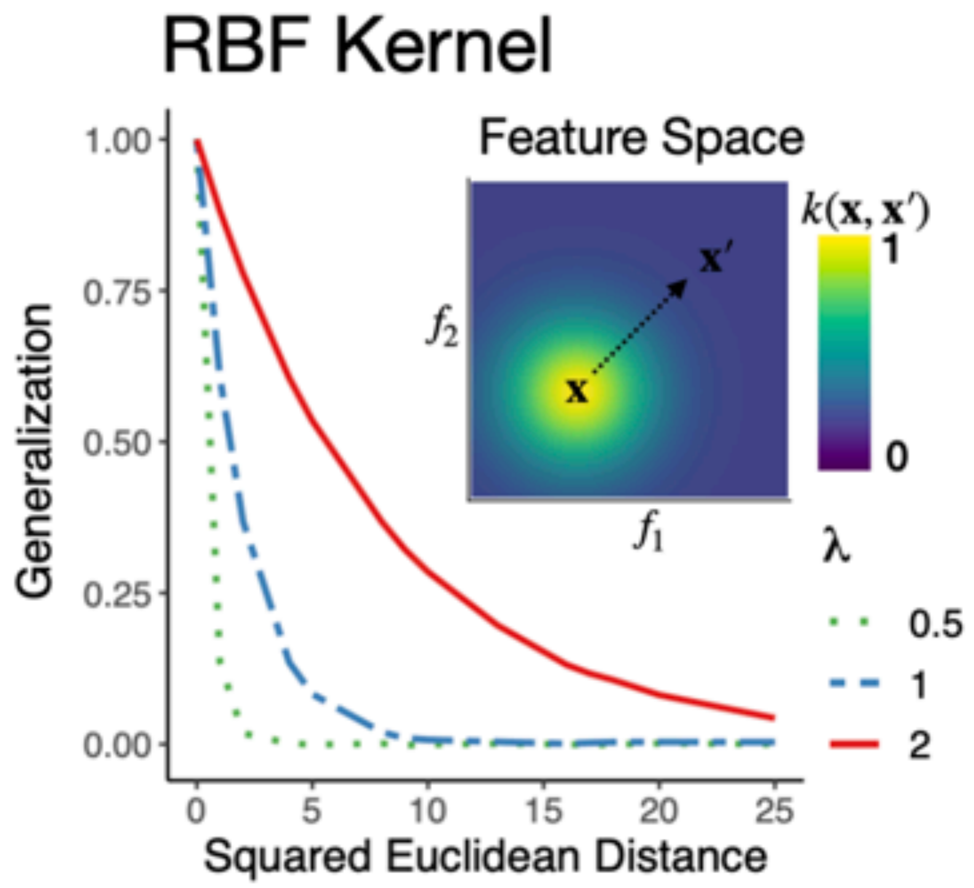
$$m(\mathbf{x}_* | \mathcal{D}) = \mathbf{k}_*^\top (\mathbf{K} + \sigma^2 \mathbf{I})^{-1} \mathbf{y}$$
$$v(\mathbf{x}_* | \mathcal{D}) = k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}_*^\top (\mathbf{K} + \sigma^2 \mathbf{I})^{-1} \mathbf{k}_*$$

see Rasmussen & Williams (2005) for further details

Gaussian Process (GP) regression in detail



- Prior over functions (i.e., hypotheses) is a multivariate Gaussian:
  $P(f) \sim \mathcal{GP}\left(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')\right)$
  - prior mean $m(\mathbf{x})$ is typically set to 0 without loss of generalization
  - Covariance $k(\mathbf{x}, \mathbf{x}')$ is defined by a choice of kernel e.g., RBF kernel:
    $$k(\mathbf{x}, \mathbf{x}') = \exp\left(\frac{-||\mathbf{x} - \mathbf{x}'||^2}{2\lambda^2}\right)$$
    where λ defines the expected smoothness of the function

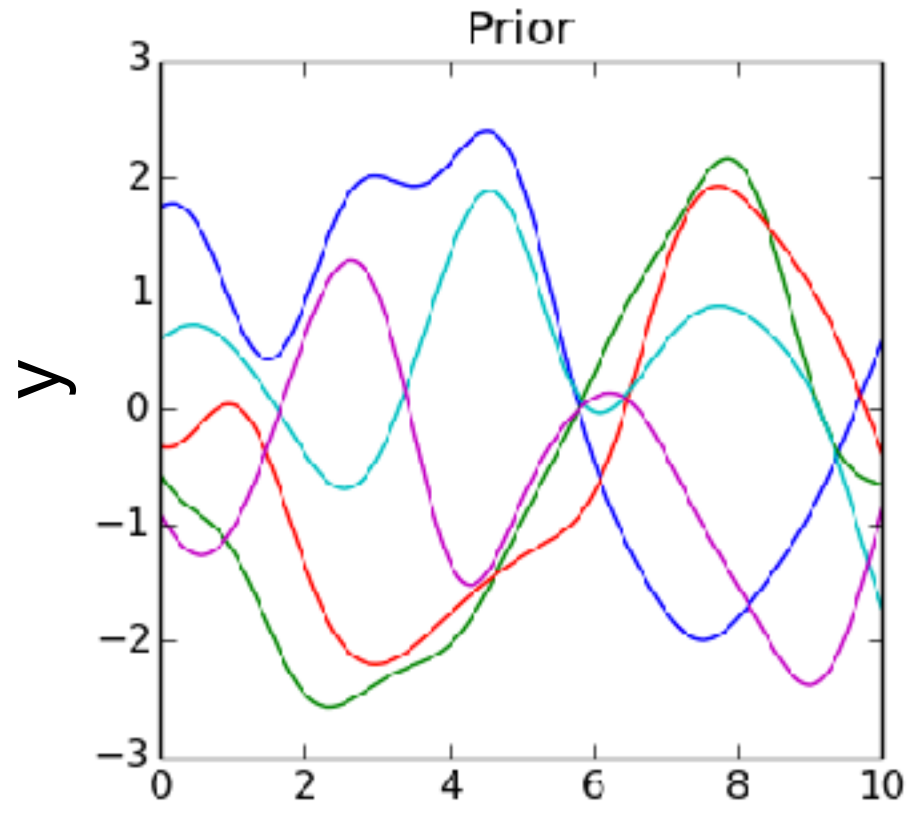


- Once we acqire some data $\mathscr{D} = \{\mathbf{X}, \mathbf{y}\}$, we can compute a posterior prediction about any new datapoint $\mathbf{x}_*$ that is also Gaussian with mean and variance defined as

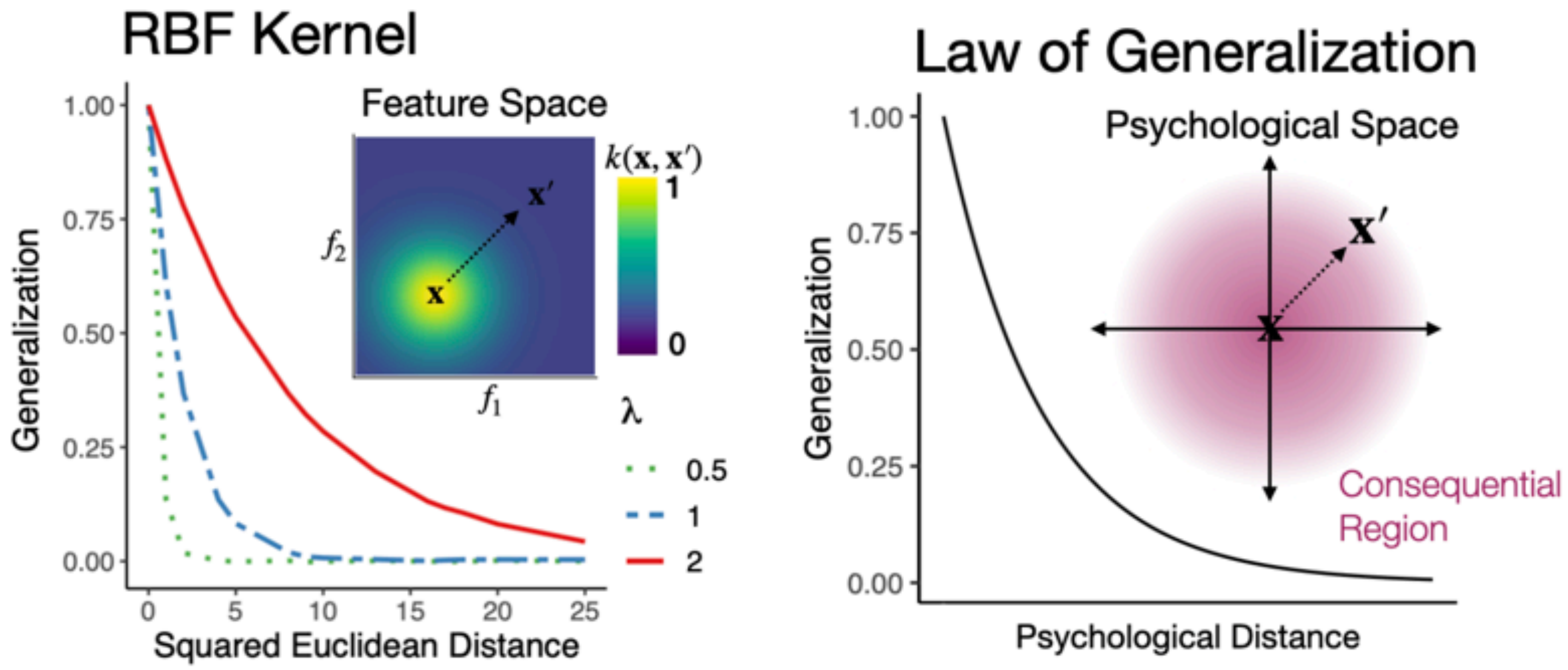
$$m(\mathbf{x}_* | \mathscr{D}) = \mathbf{k}_*^\top (\mathbf{K} + \sigma^2 \mathbf{I})^{-1} \mathbf{y}$$
$$v(\mathbf{x}_* | \mathscr{D}) = k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}_*^\top (\mathbf{K} + \sigma^2 \mathbf{I})^{-1} \mathbf{k}_*$$

see Rasmussen & Williams (2005) for further details

Gaussian Process (GP) regression in detail



- Prior over functions (i.e., hypotheses) is a multivariate Gaussian:
  $P(f) \sim \mathcal{GP}\left(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')\right)$
  - prior mean $m(\mathbf{x})$ is typically set to 0 without loss of generalization
  - Covariance $k(\mathbf{x}, \mathbf{x}')$ is defined by a choice of kernel e.g., RBF kernel:
    $$k(\mathbf{x}, \mathbf{x}') = \exp\left(\frac{-||\mathbf{x} - \mathbf{x}'||^2}{2\lambda^2}\right)$$
    where λ defines the expected smoothness of the function

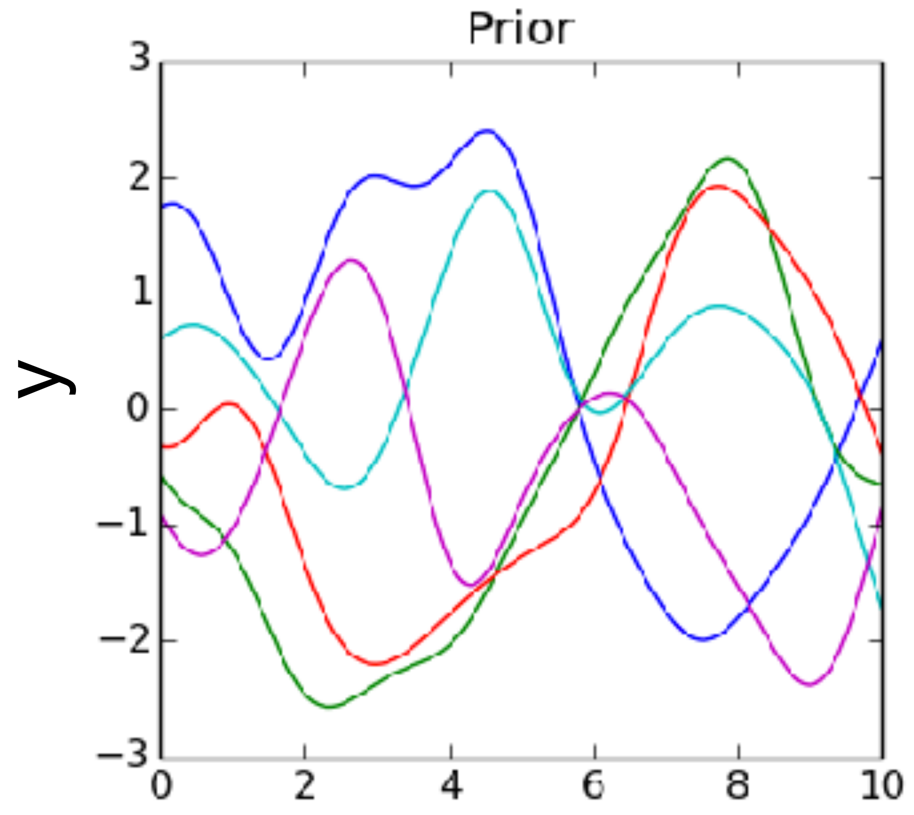


- Once we acqire some data $\mathcal{D} = \{\mathbf{X}, \mathbf{y}\}$, we can compute a posterior prediction about any new datapoint $\mathbf{x}_*$ that is also Gaussian with mean and variance defined as

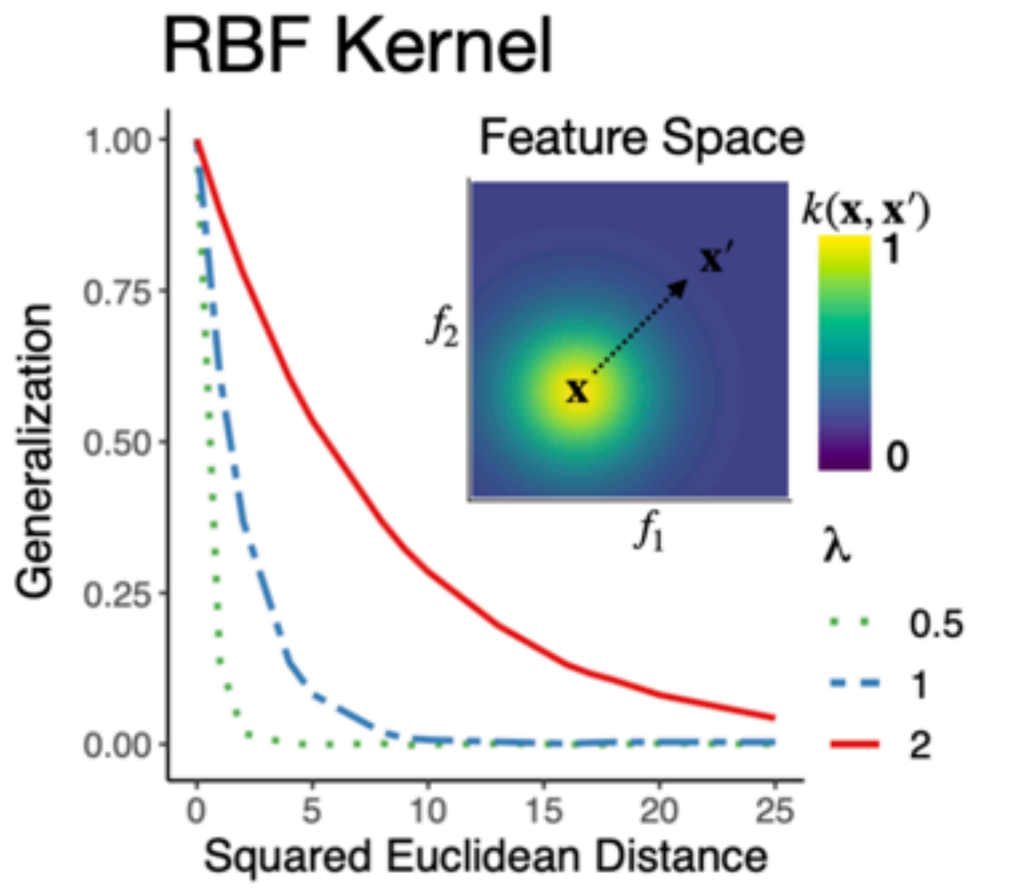
$$m(\mathbf{x}_* | \mathcal{D}) = \mathbf{k}_*^\top (\mathbf{K} + \sigma^2 \mathbf{I})^{-1} \mathbf{y}$$
$$v(\mathbf{x}_* | \mathcal{D}) = k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}_*^\top (\mathbf{K} + \sigma^2 \mathbf{I})^{-1} \mathbf{k}_*$$

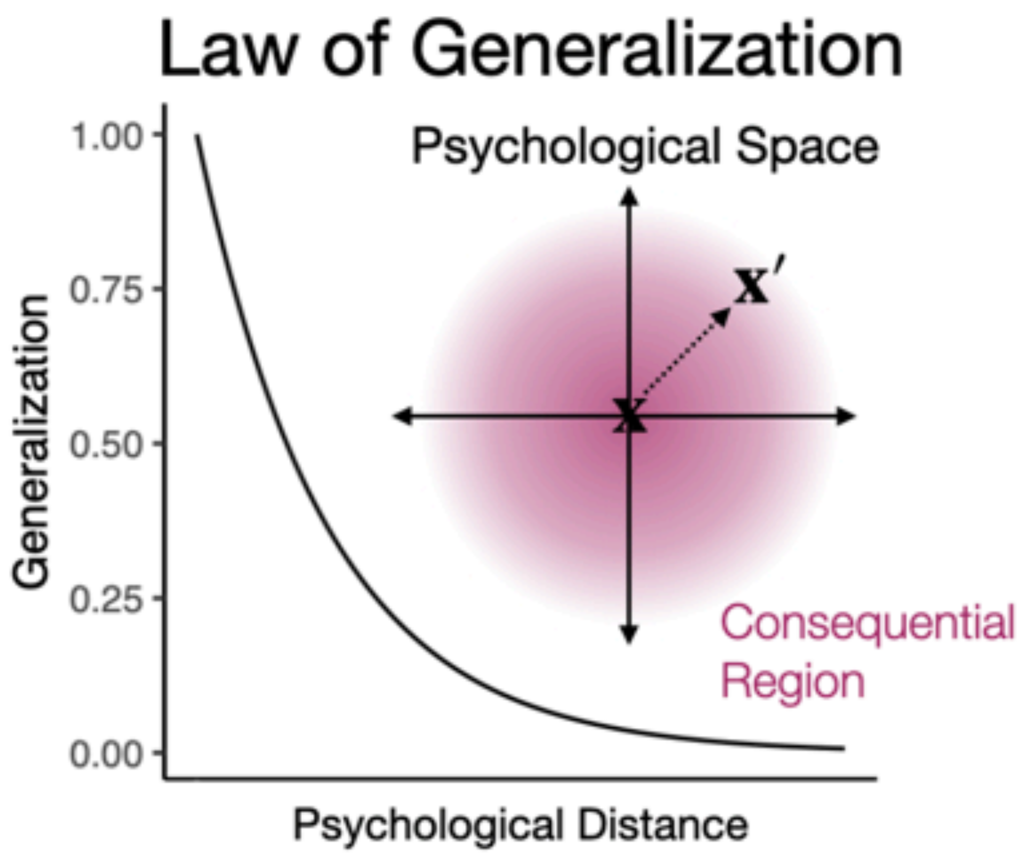
see Rasmussen & Williams (2005) for further details

Gaussian Process (GP) regression in detail



- Prior over functions (i.e., hypotheses) is a multivariate Gaussian:
  $P(f) \sim \mathcal{GP}\left(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')\right)$
  - prior mean $m(\mathbf{x})$ is typically set to 0 without loss of generalization
  - Covariance $k(\mathbf{x}, \mathbf{x}')$ is defined by a choice of kernel e.g., RBF kernel:
    $$k(\mathbf{x}, \mathbf{x}') = \exp\left(\frac{-||\mathbf{x} - \mathbf{x}'||^2}{2\lambda^2}\right)$$
    where λ defines the expected smoothness of the function

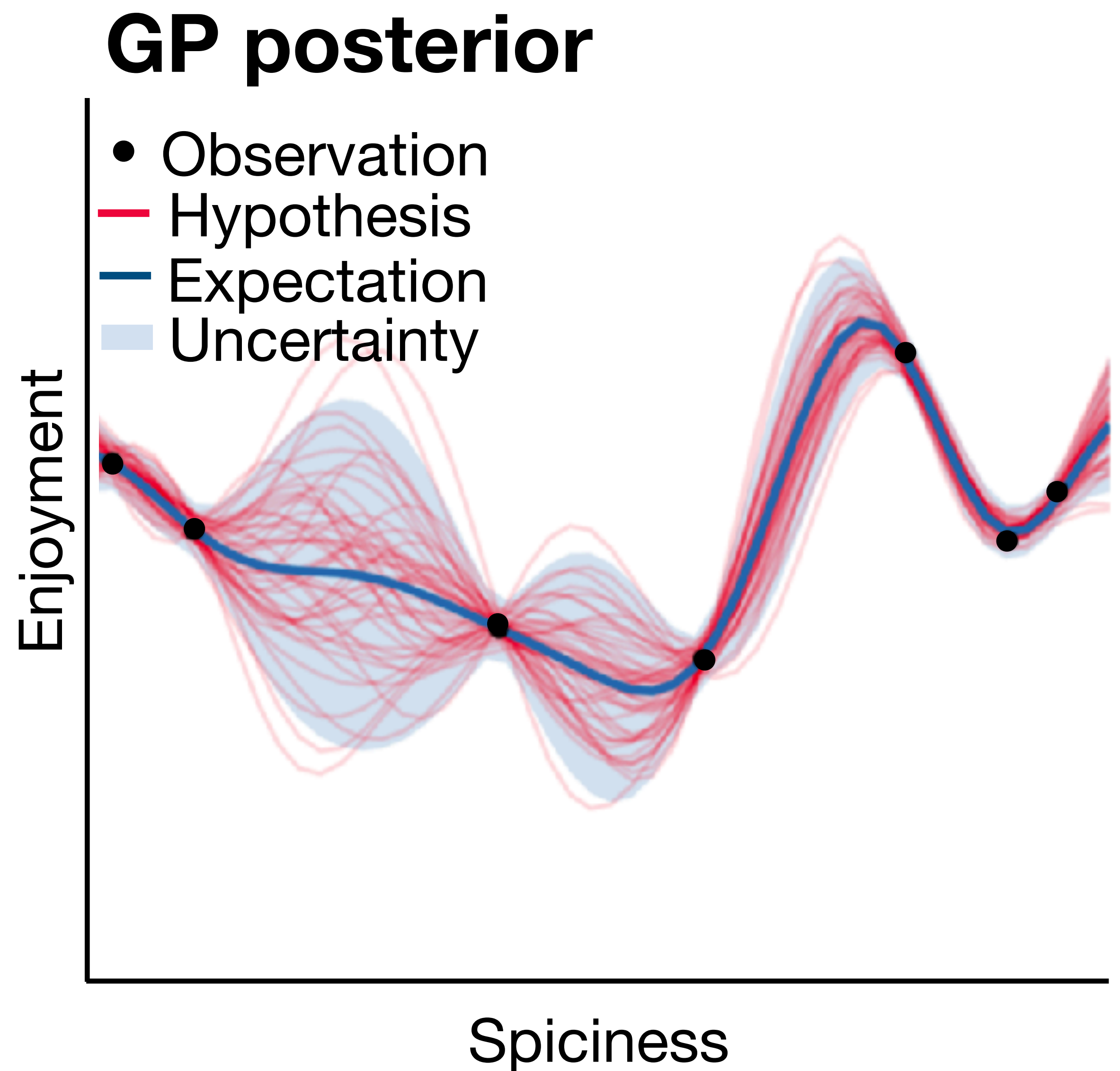




- Once we acqire some data $\mathscr{D} = \{\mathbf{X}, \mathbf{y}\}$, we can compute a posterior prediction about any new datapoint $\mathbf{x}_*$ that is also Gaussian with mean and variance defined as

$$m(\mathbf{x}_* | \mathscr{D}) = \mathbf{k}_*^\top (\mathbf{K} + \sigma^2 \mathbf{I})^{-1} \mathbf{y}$$

$$v(\mathbf{x}_* | \mathscr{D}) = k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}_*^\top (\mathbf{K} + \sigma^2 \mathbf{I})^{-1} \mathbf{k}_*$$



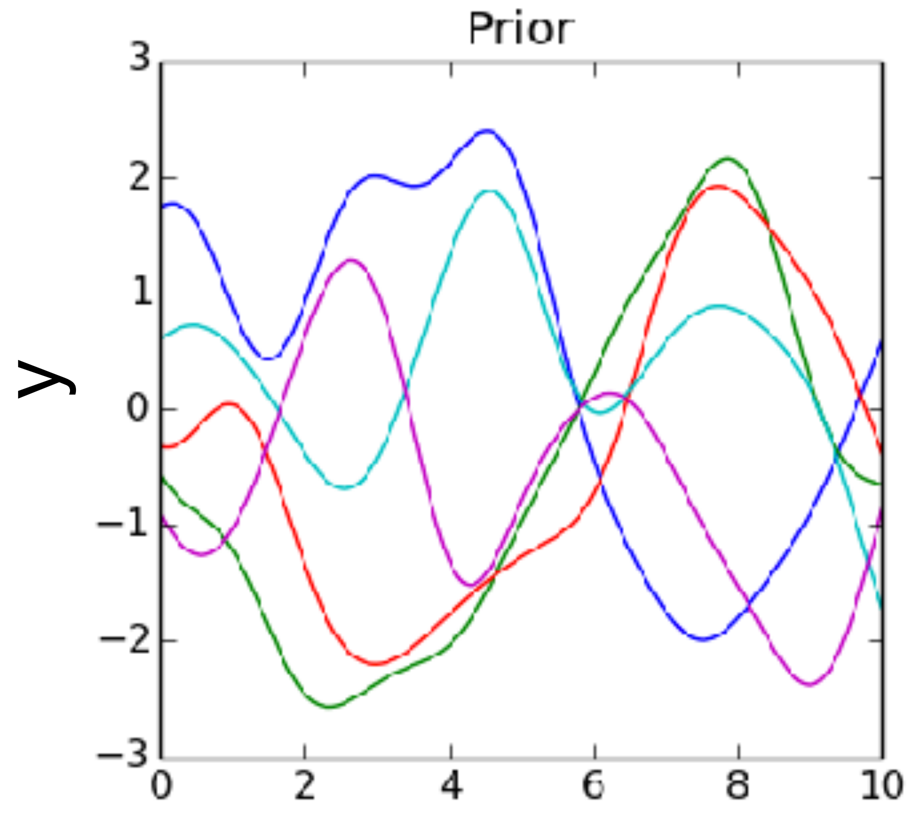
see Rasmussen & Williams (2005) for further details

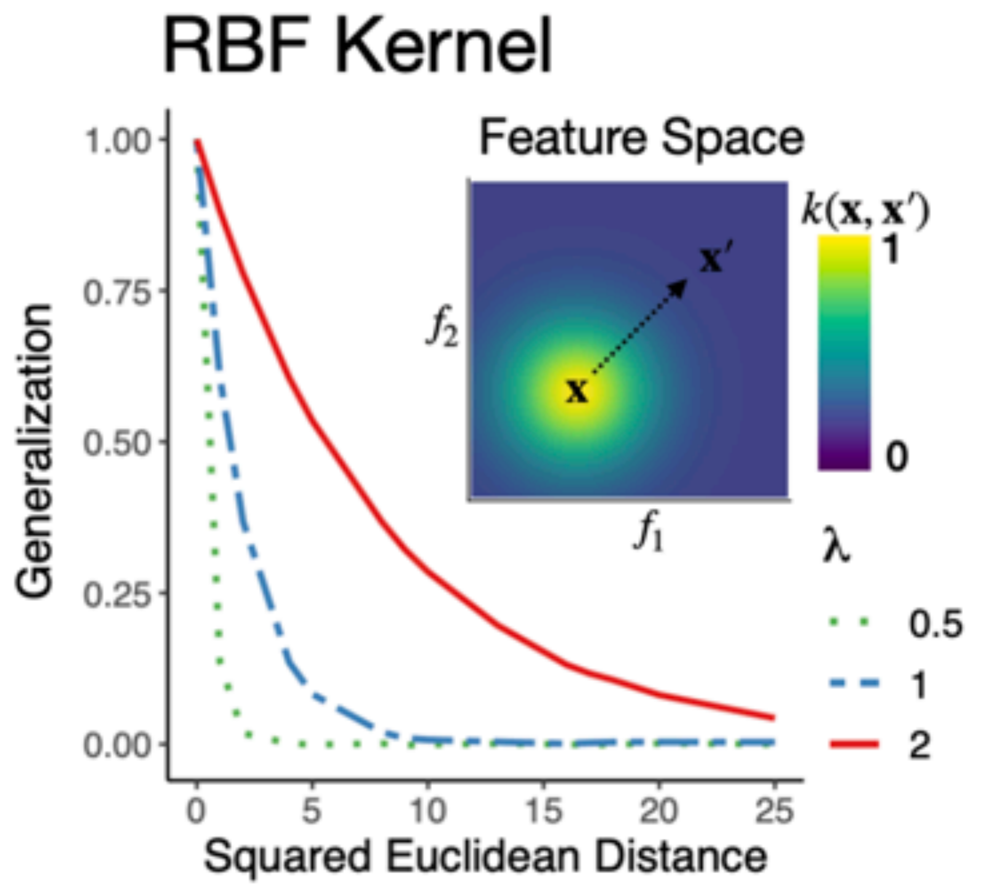
Gaussian Process (GP) regression in detail

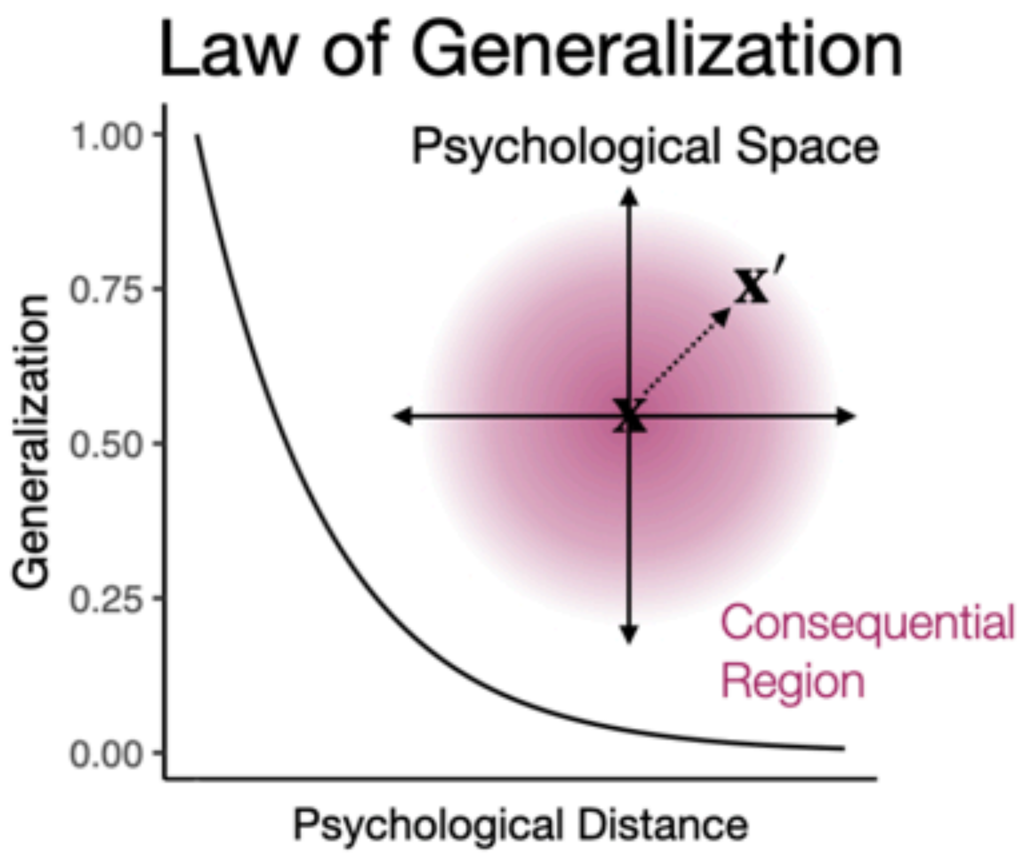
- Prior over functions (i.e., hypotheses) is a multivariate Gaussian:
  $P(f) \sim \mathcal{GP}\left(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')\right)$
  - prior mean $m(\mathbf{x})$ is typically set to 0 without loss of generalization
  - Covariance $k(\mathbf{x}, \mathbf{x}')$ is defined by a choice of kernel e.g., RBF kernel:
    $$k(\mathbf{x}, \mathbf{x}') = \exp\left(\frac{-||\mathbf{x} - \mathbf{x}'||^2}{2\lambda^2}\right)$$
    where λ defines the expected smoothness of the function
- Once we acqire some data $\mathcal{D} = \{\mathbf{X}, \mathbf{y}\}$, we can compute a posterior prediction about any new datapoint $\mathbf{x}_*$ that is also Gaussian with mean and variance defined as

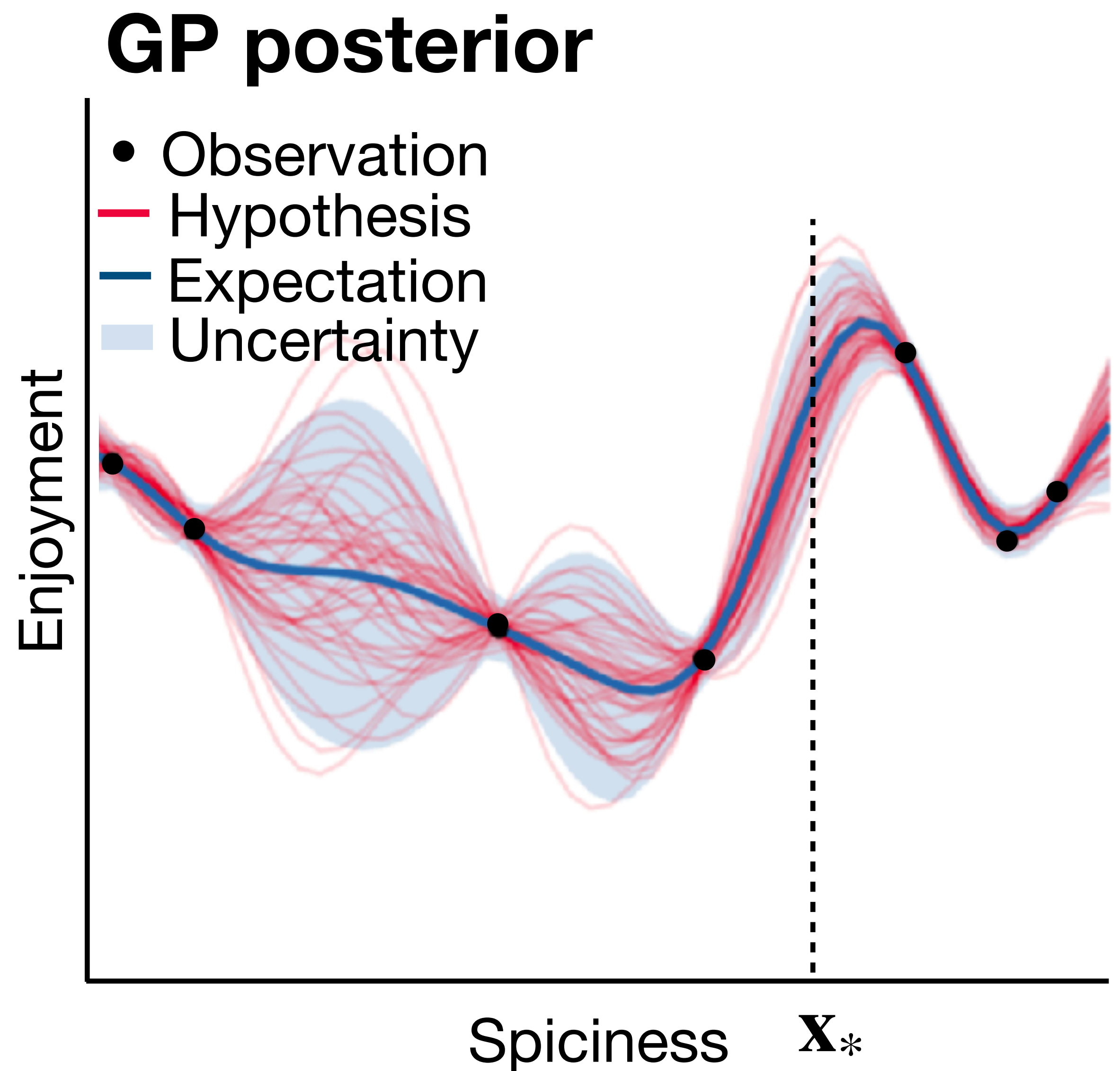
$$m(\mathbf{x}_* | \mathcal{D}) = \mathbf{k}_*^\top (\mathbf{K} + \sigma^2 \mathbf{I})^{-1} \mathbf{y}$$

$$v(\mathbf{x}_* | \mathcal{D}) = k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}_*^\top (\mathbf{K} + \sigma^2 \mathbf{I})^{-1} \mathbf{k}_*$$



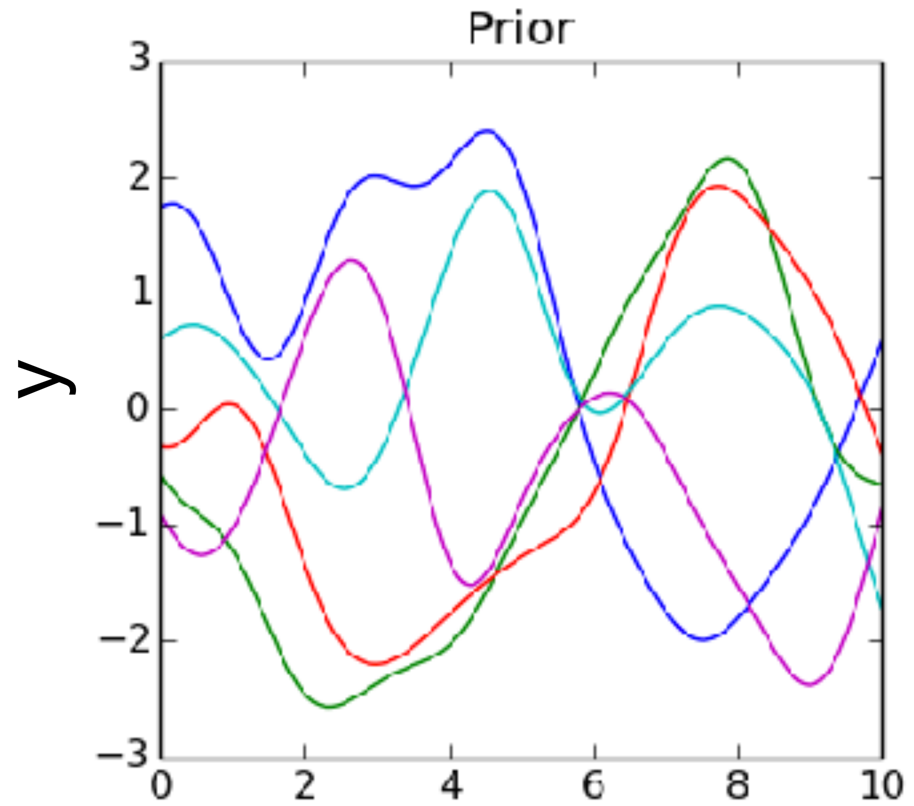






see Rasmussen & Williams (2005) for further details

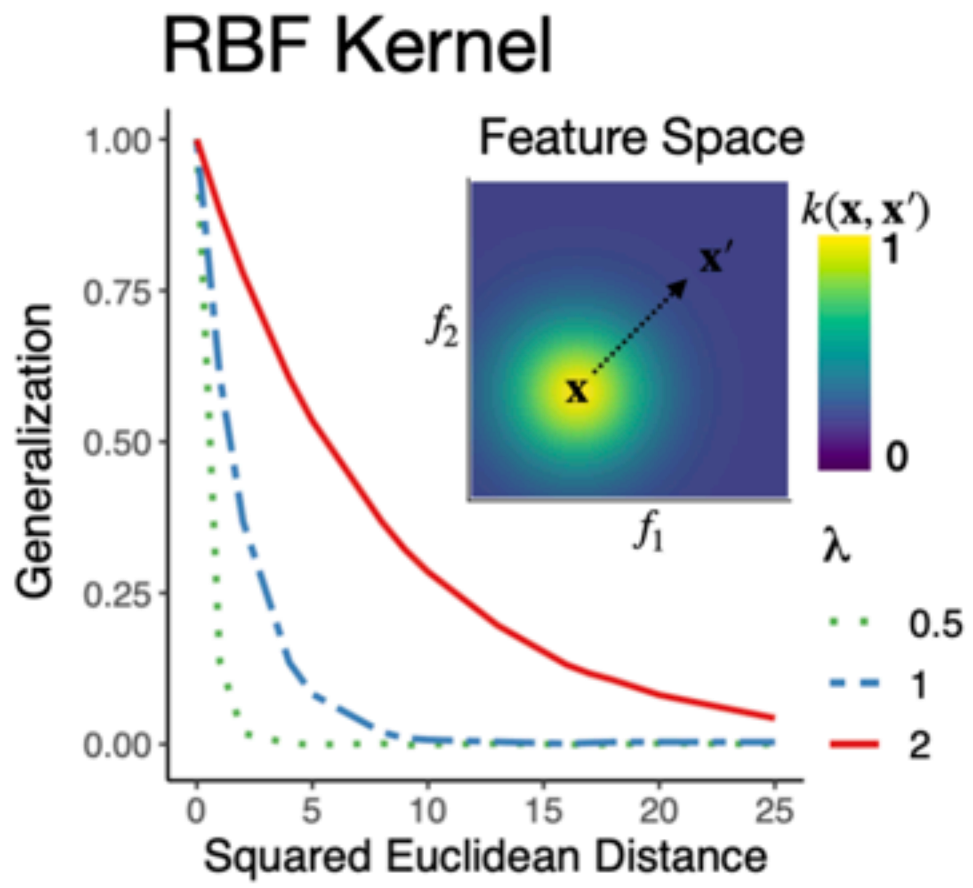
Gaussian Process (GP) regression in detail

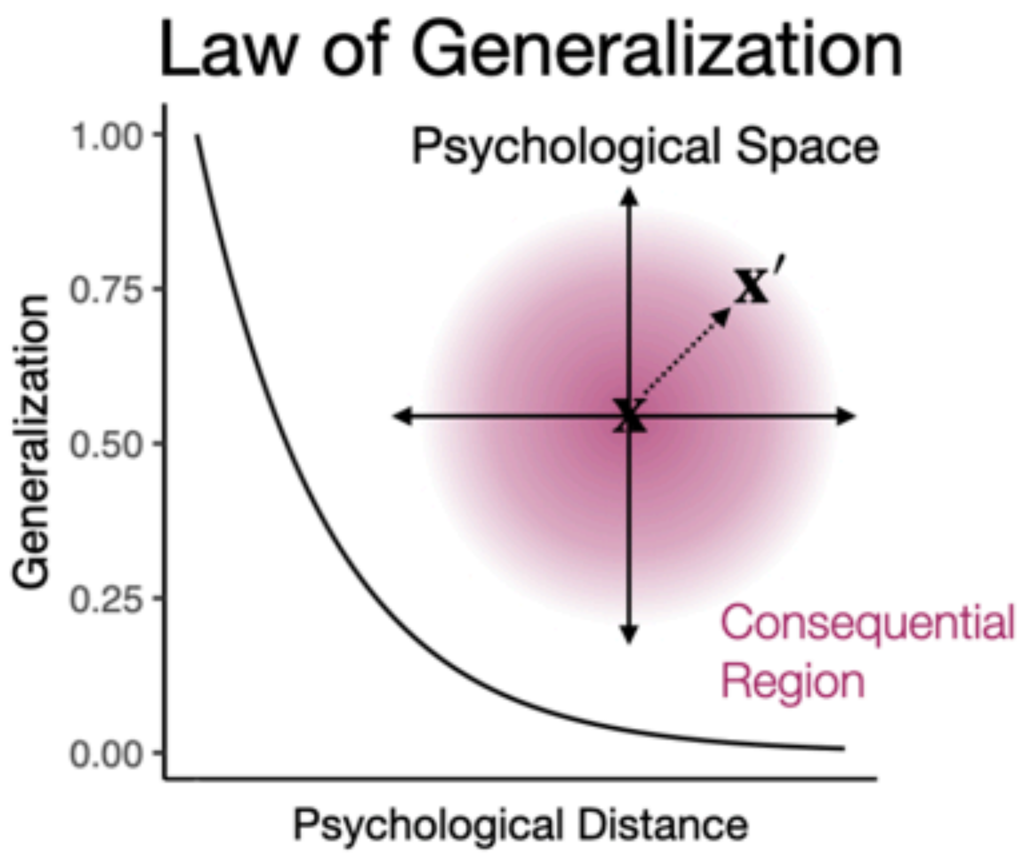


- Prior over functions (i.e., hypotheses) is a multivariate Gaussian:
  $P(f) \sim \mathcal{GP}\left(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')\right)$
  - prior mean $m(\mathbf{x})$ is typically set to 0 without loss of generalization
  - Covariance $k(\mathbf{x}, \mathbf{x}')$ is defined by a choice of kernel e.g., RBF kernel:

$$k(\mathbf{x}, \mathbf{x}') = \exp\left(\frac{-||\mathbf{x} - \mathbf{x}'||^2}{2\lambda^2}\right)$$

    where λ defines the expected smoothness of the function

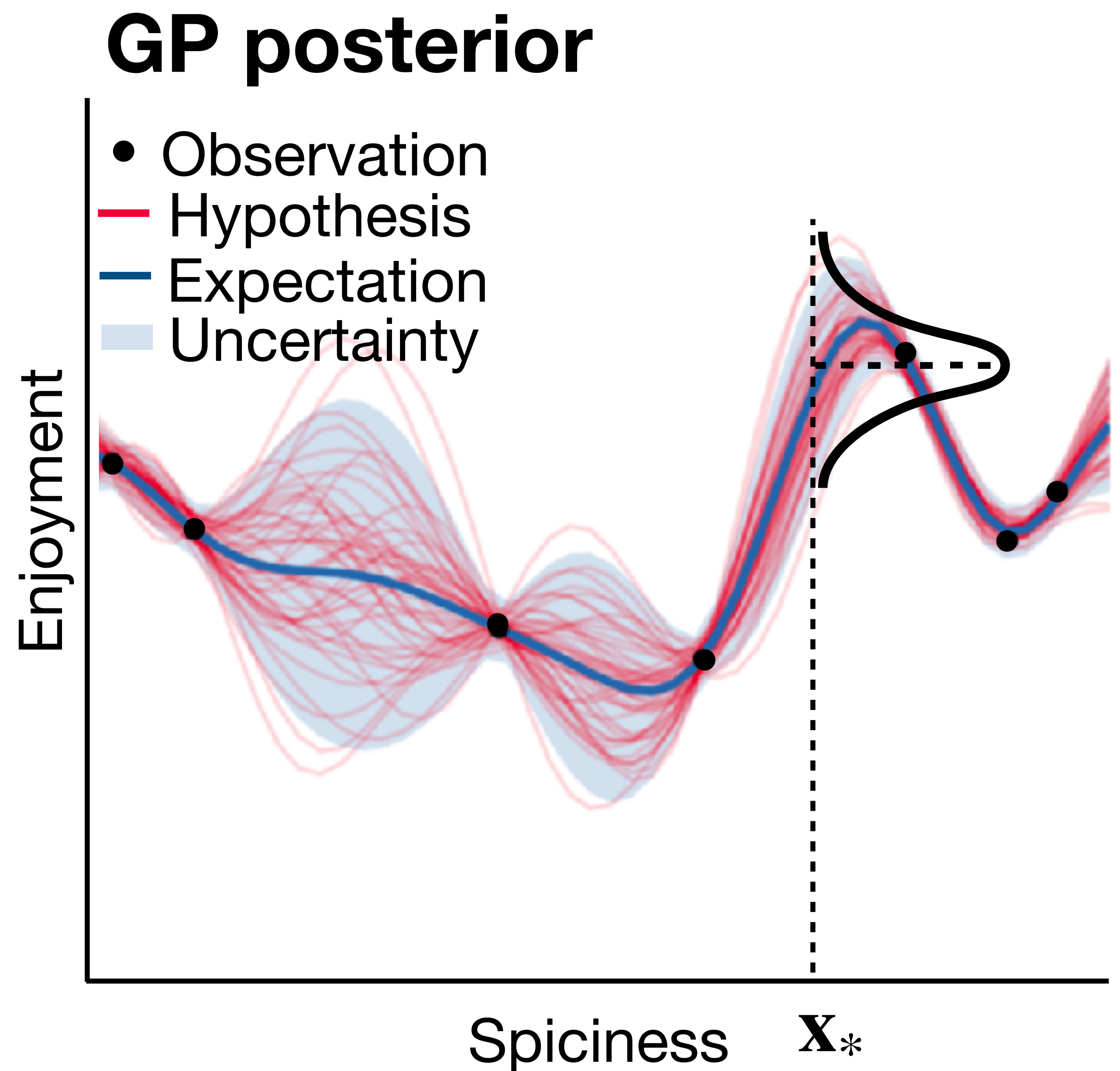




- Once we acqire some data $\mathcal{D} = \{\mathbf{X}, \mathbf{y}\}$, we can compute a posterior prediction about any new datapoint $\mathbf{x}_*$ that is also Gaussian with mean and variance defined as

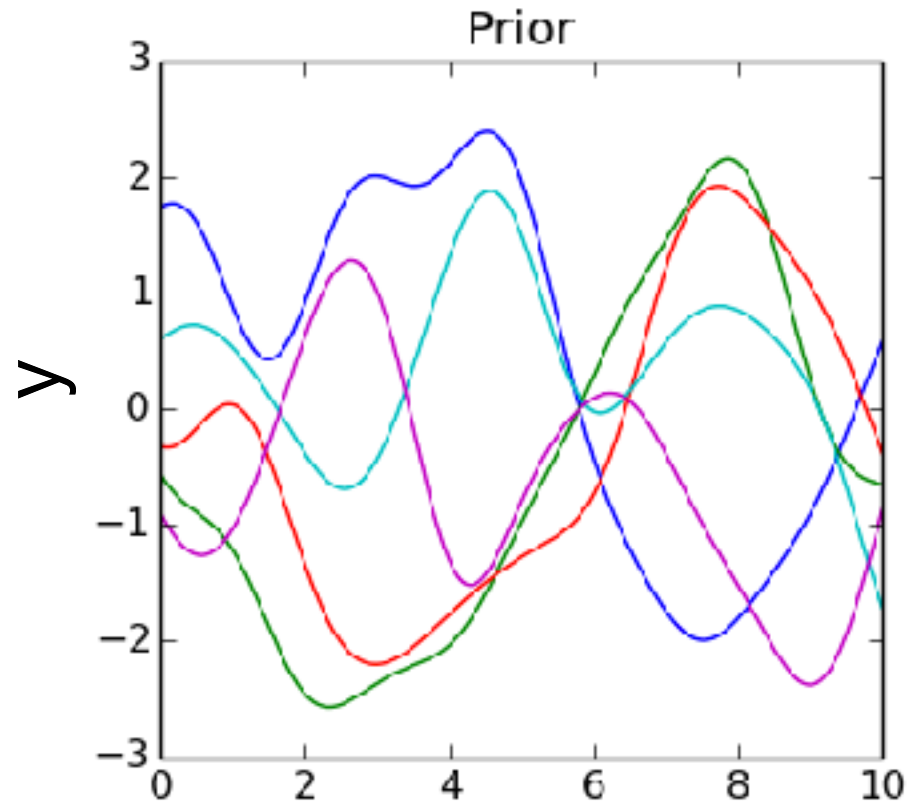
$$m(\mathbf{x}_* | \mathcal{D}) = \mathbf{k}_*^\top (\mathbf{K} + \sigma^2 \mathbf{I})^{-1} \mathbf{y}$$

$$v(\mathbf{x}_* | \mathcal{D}) = k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}_*^\top (\mathbf{K} + \sigma^2 \mathbf{I})^{-1} \mathbf{k}_*$$

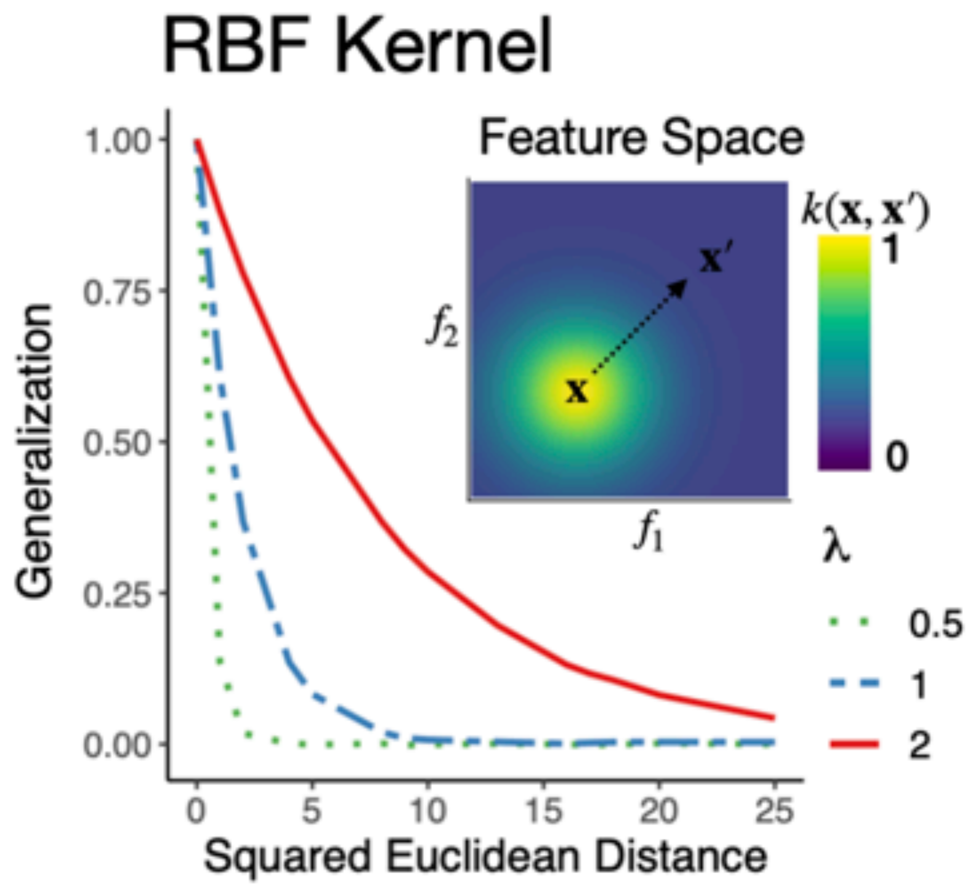


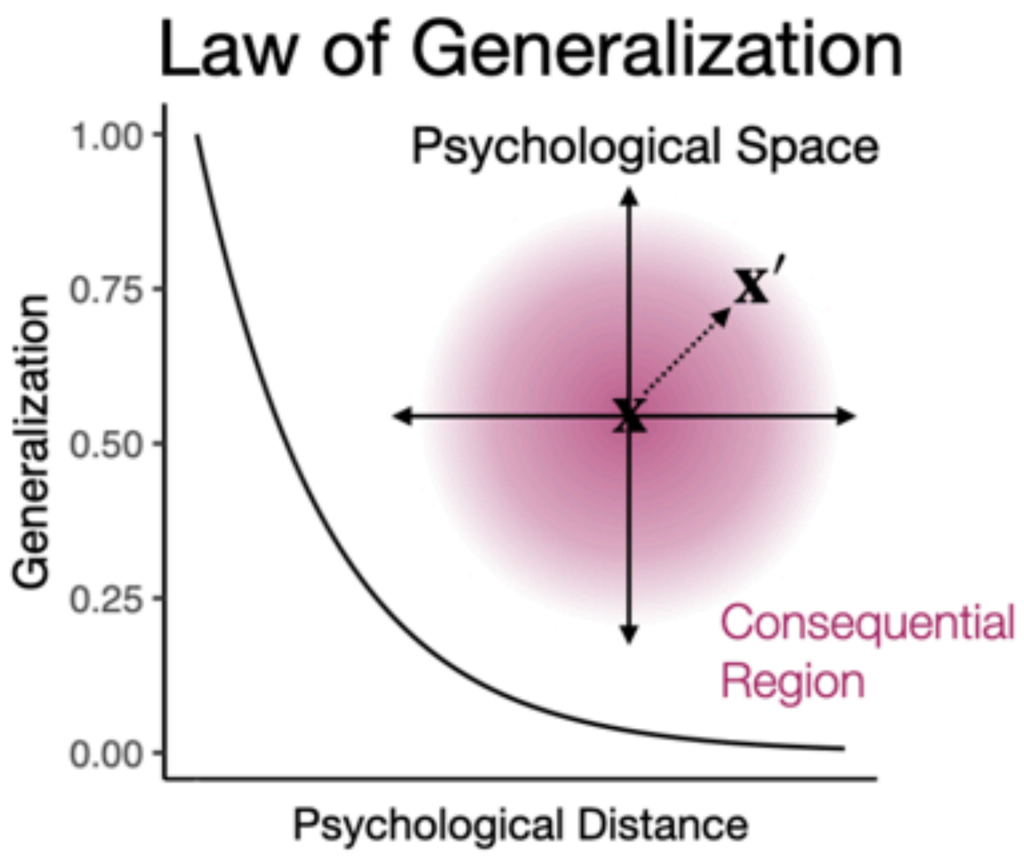
see Rasmussen & Williams (2005) for further details

Gaussian Process (GP) regression in detail



- Prior over functions (i.e., hypotheses) is a multivariate Gaussian:
  $P(f) \sim \mathcal{GP}\left(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')\right)$
  - prior mean $m(\mathbf{x})$ is typically set to 0 without loss of generalization
  - Covariance $k(\mathbf{x}, \mathbf{x}')$ is defined by a choice of kernel e.g., RBF kernel:
    $$k(\mathbf{x}, \mathbf{x}') = \exp\left(\frac{-||\mathbf{x} - \mathbf{x}'||^2}{2\lambda^2}\right)$$
    where λ defines the expected smoothness of the function

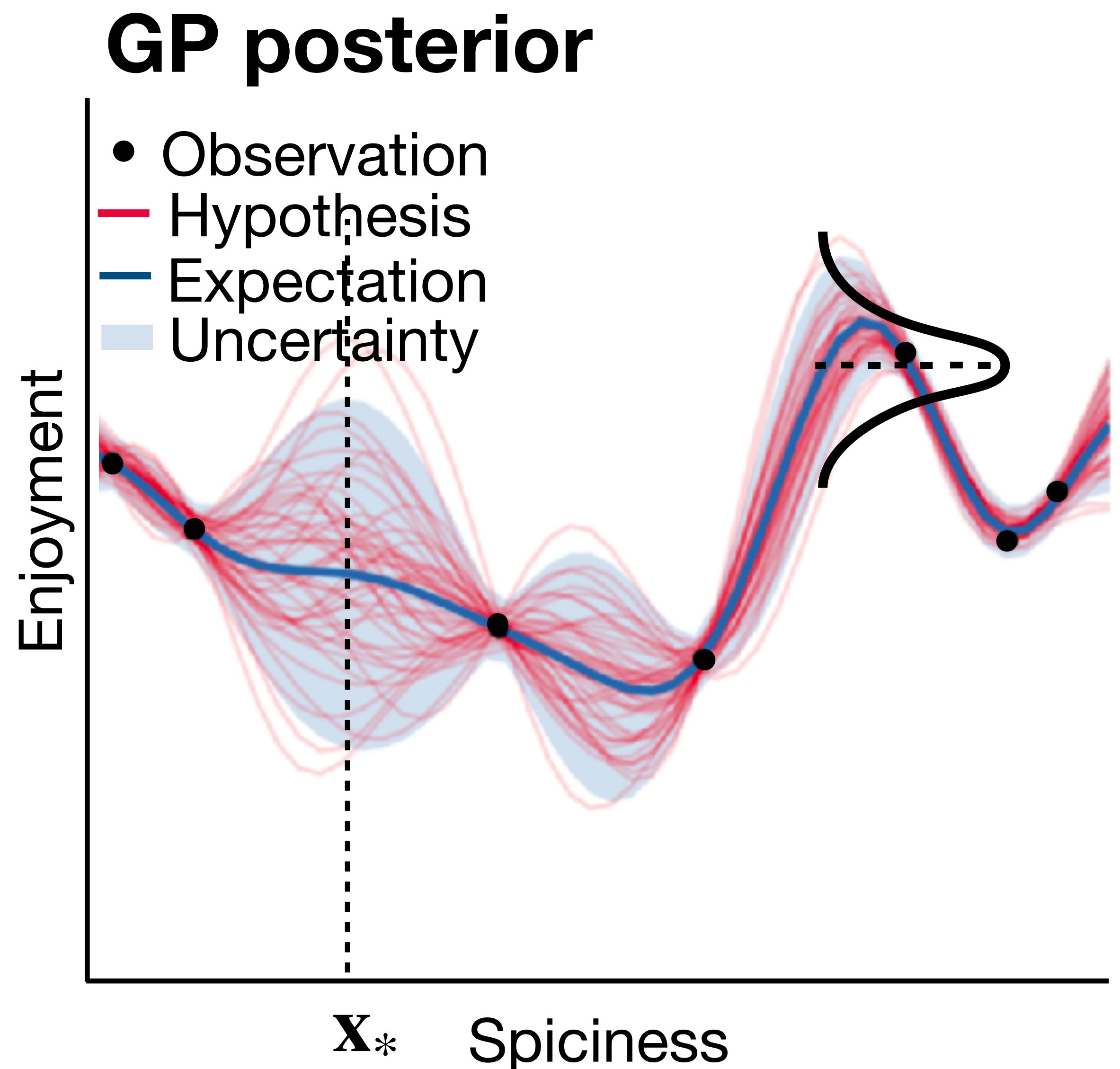




- Once we acqire some data $\mathcal{D} = \{\mathbf{X}, \mathbf{y}\}$, we can compute a posterior prediction about any new datapoint $\mathbf{x}_*$ that is also Gaussian with mean and variance defined as

$$m(\mathbf{x}_* | \mathcal{D}) = \mathbf{k}_*^\top(\mathbf{K} + \sigma^2\mathbf{I})^{-1}\mathbf{y}$$

$$v(\mathbf{x}_* | \mathcal{D}) = k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}_*^\top(\mathbf{K} + \sigma^2\mathbf{I})^{-1}\mathbf{k}_*$$



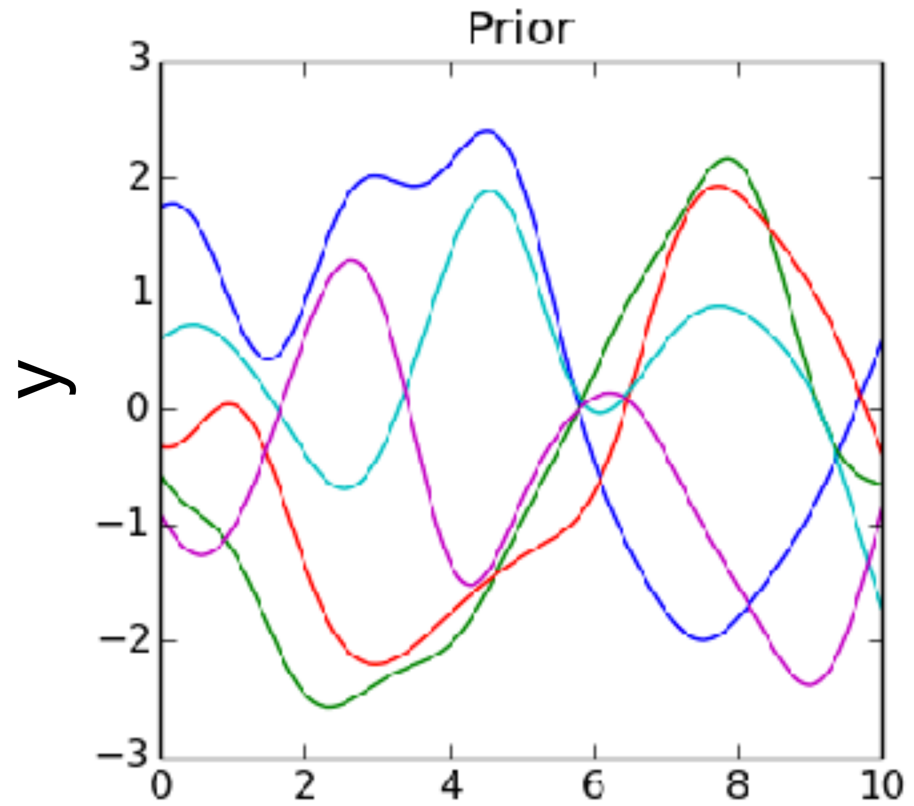
see Rasmussen & Williams (2005) for further details

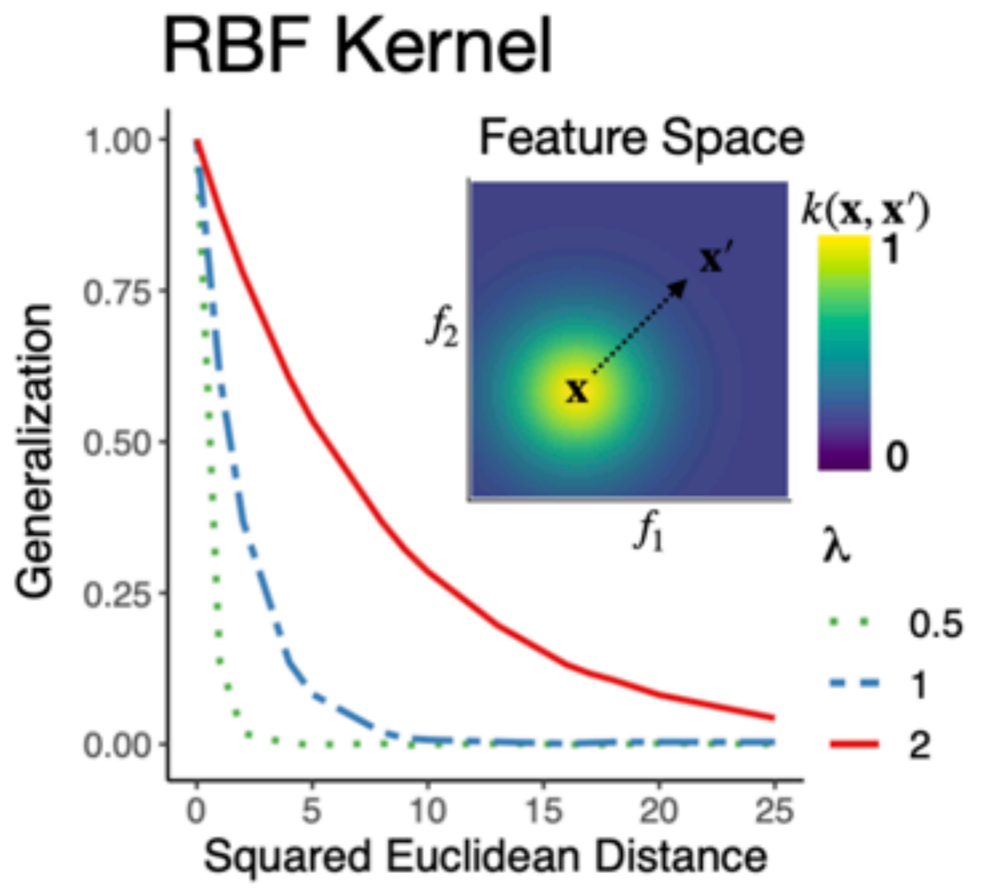
Gaussian Process (GP) regression in detail

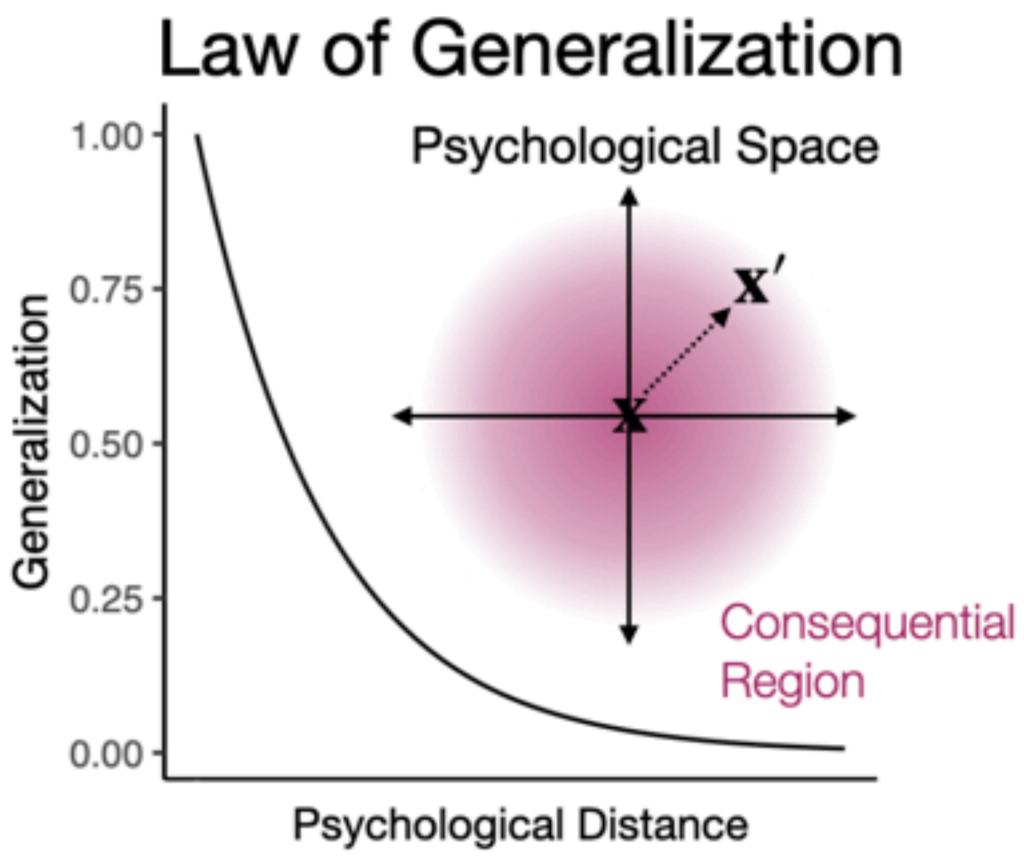
- Prior over functions (i.e., hypotheses) is a multivariate Gaussian:
  $P(f) \sim \mathcal{GP}\left(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')\right)$
  - prior mean $m(\mathbf{x})$ is typically set to 0 without loss of generalization
  - Covariance $k(\mathbf{x}, \mathbf{x}')$ is defined by a choice of kernel e.g., RBF kernel:
    $$k(\mathbf{x}, \mathbf{x}') = \exp\left(\frac{-||\mathbf{x}-\mathbf{x}'||^2}{2\lambda^2}\right)$$
    where λ defines the expected smoothness of the function
- Once we acqire some data $\mathcal{D} = \{\mathbf{X}, \mathbf{y}\}$, we can compute a posterior prediction about any new datapoint $\mathbf{x}_*$ that is also Gaussian with mean and variance defined as

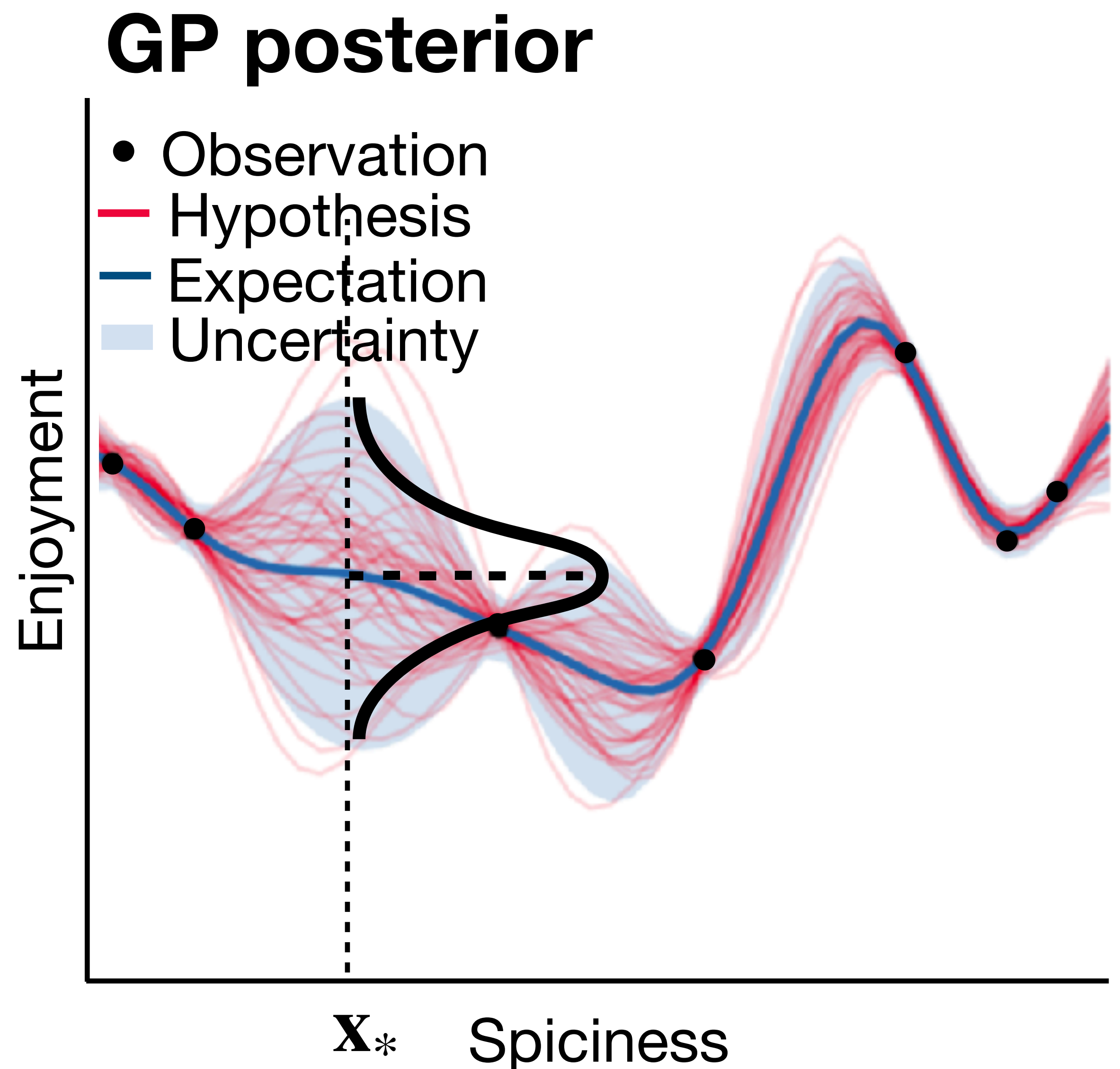
$$m(\mathbf{x}_* | \mathcal{D}) = \mathbf{k}_*^\top (\mathbf{K} + \sigma^2 \mathbf{I})^{-1} \mathbf{y}$$

$$v(\mathbf{x}_* | \mathcal{D}) = k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}_*^\top (\mathbf{K} + \sigma^2 \mathbf{I})^{-1} \mathbf{k}_*$$



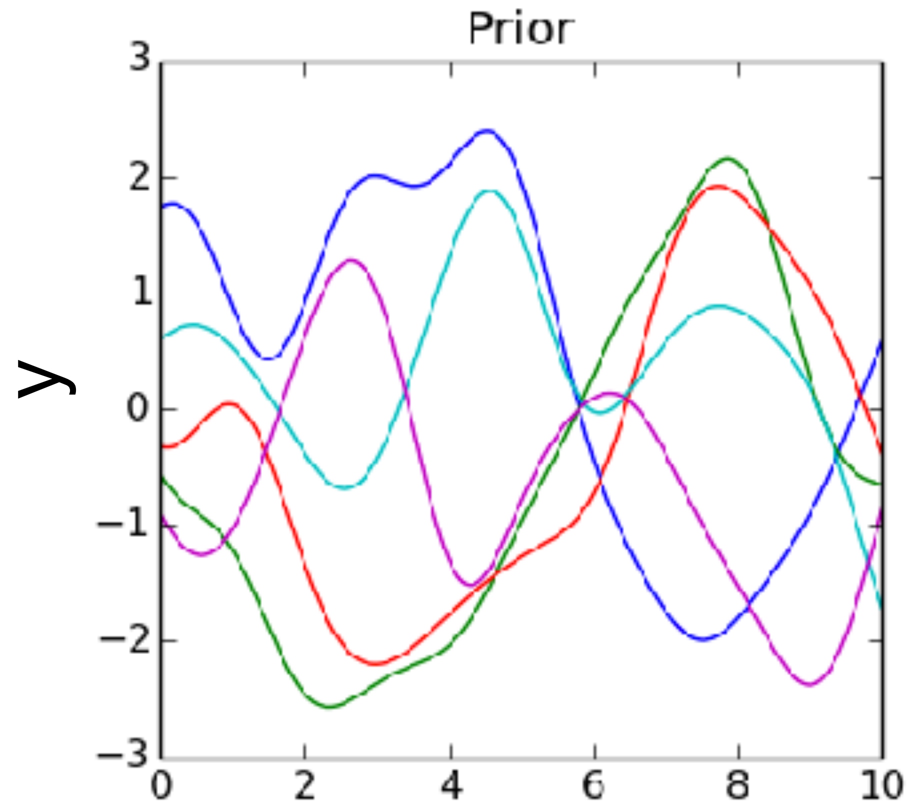




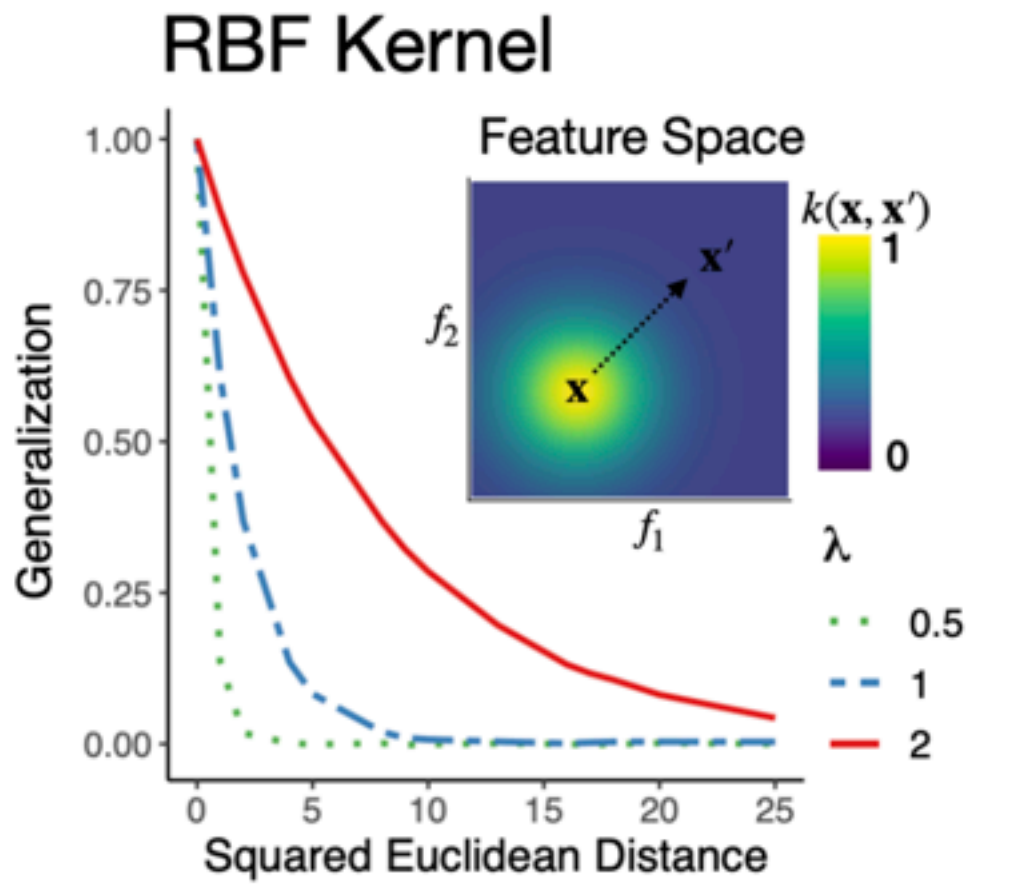


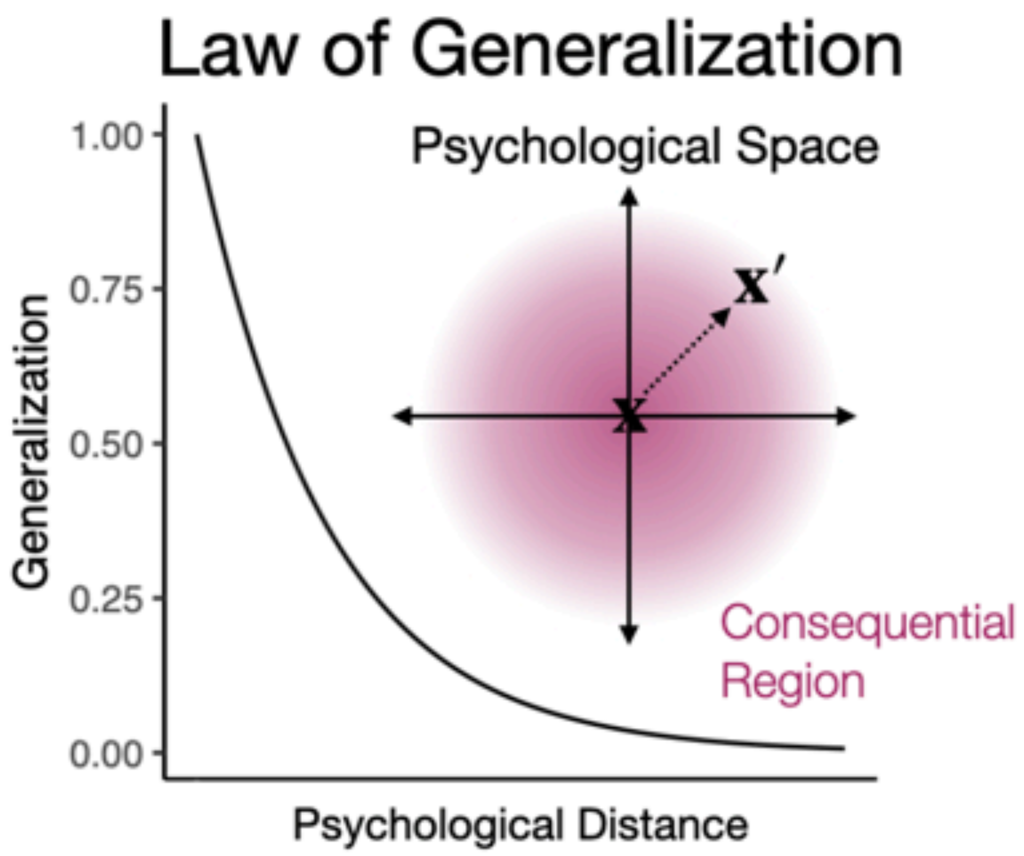
see Rasmussen & Williams (2005) for further details

Gaussian Process (GP) regression in detail



- Prior over functions (i.e., hypotheses) is a multivariate Gaussian:
  $P(f) \sim \mathcal{GP}\left(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')\right)$
    - prior mean $m(\mathbf{x})$ is typically set to 0 without loss of generalization
    - Covariance $k(\mathbf{x}, \mathbf{x}')$ is defined by a choice of kernel e.g., RBF kernel:
      $$k(\mathbf{x}, \mathbf{x}') = \exp\left(\frac{-||\mathbf{x} - \mathbf{x}'||^2}{2\lambda^2}\right)$$
      where λ defines the expected smoothness of the function



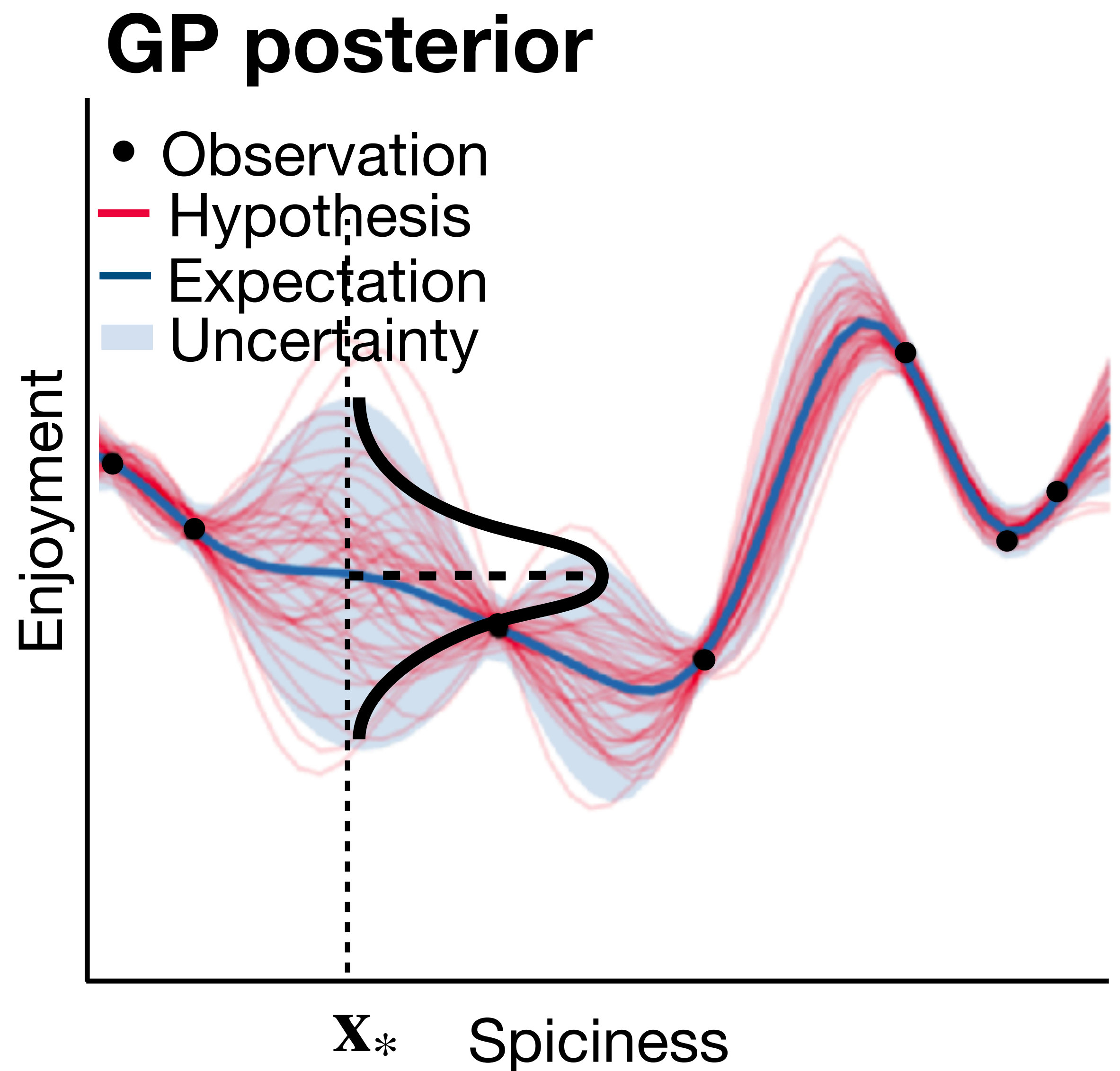


- Once we acqire some data $\mathscr{D} = \{\mathbf{X}, \mathbf{y}\}$, we can compute a posterior prediction about any new datapoint $\mathbf{x}_*$ that is also Gaussian with mean and variance defined as

$$m(\mathbf{x}_* | \mathscr{D}) = \mathbf{k}_*^\top(\mathbf{K} + \sigma^2\mathbf{I})^{-1}\mathbf{y}$$

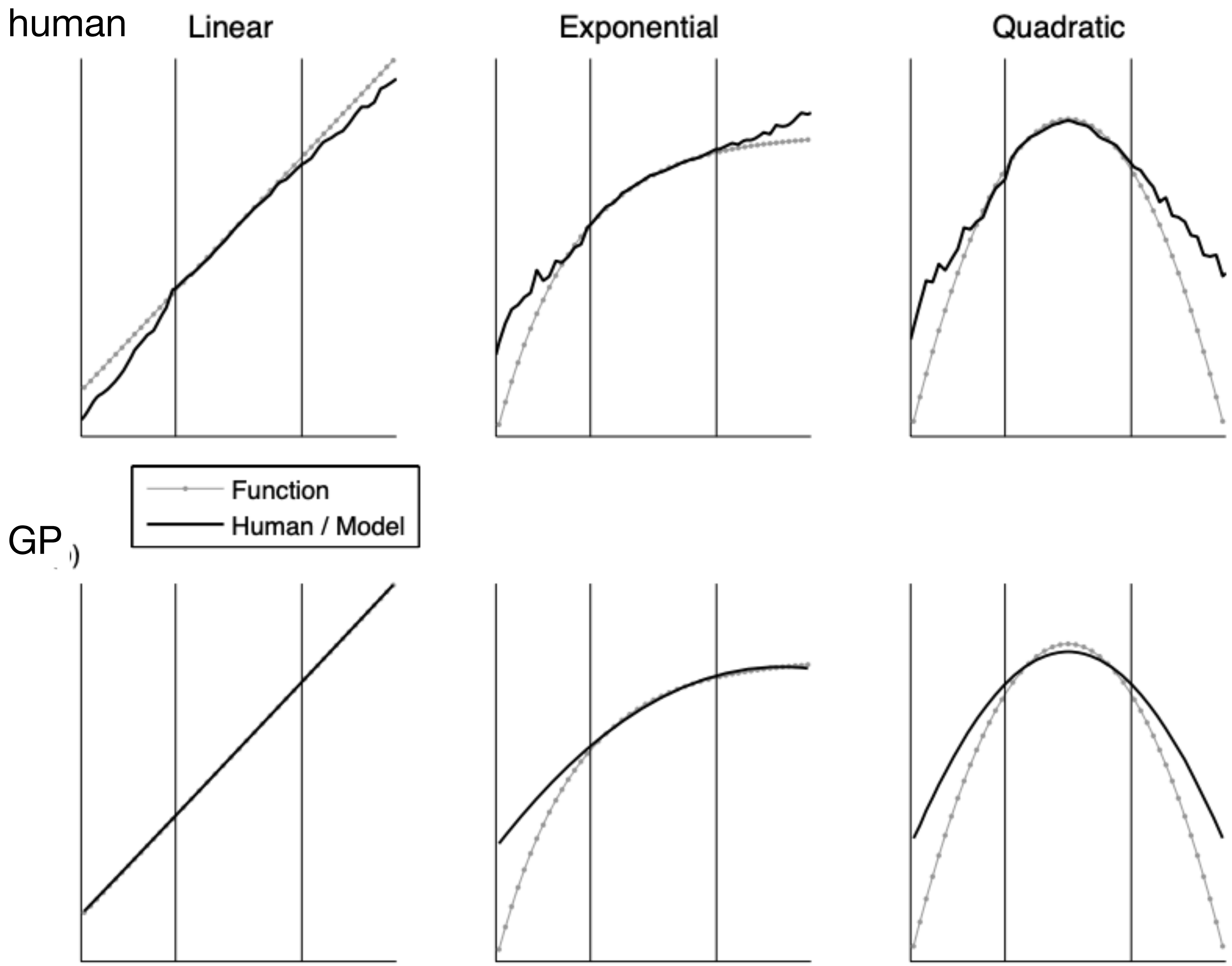
$$v(\mathbf{x}_* | \mathscr{D}) = k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}_*^\top(\mathbf{K} + \sigma^2\mathbf{I})^{-1}\mathbf{k}_*$$

*Don't worry too much about what these equations mean for now; I will provide some intuitions later



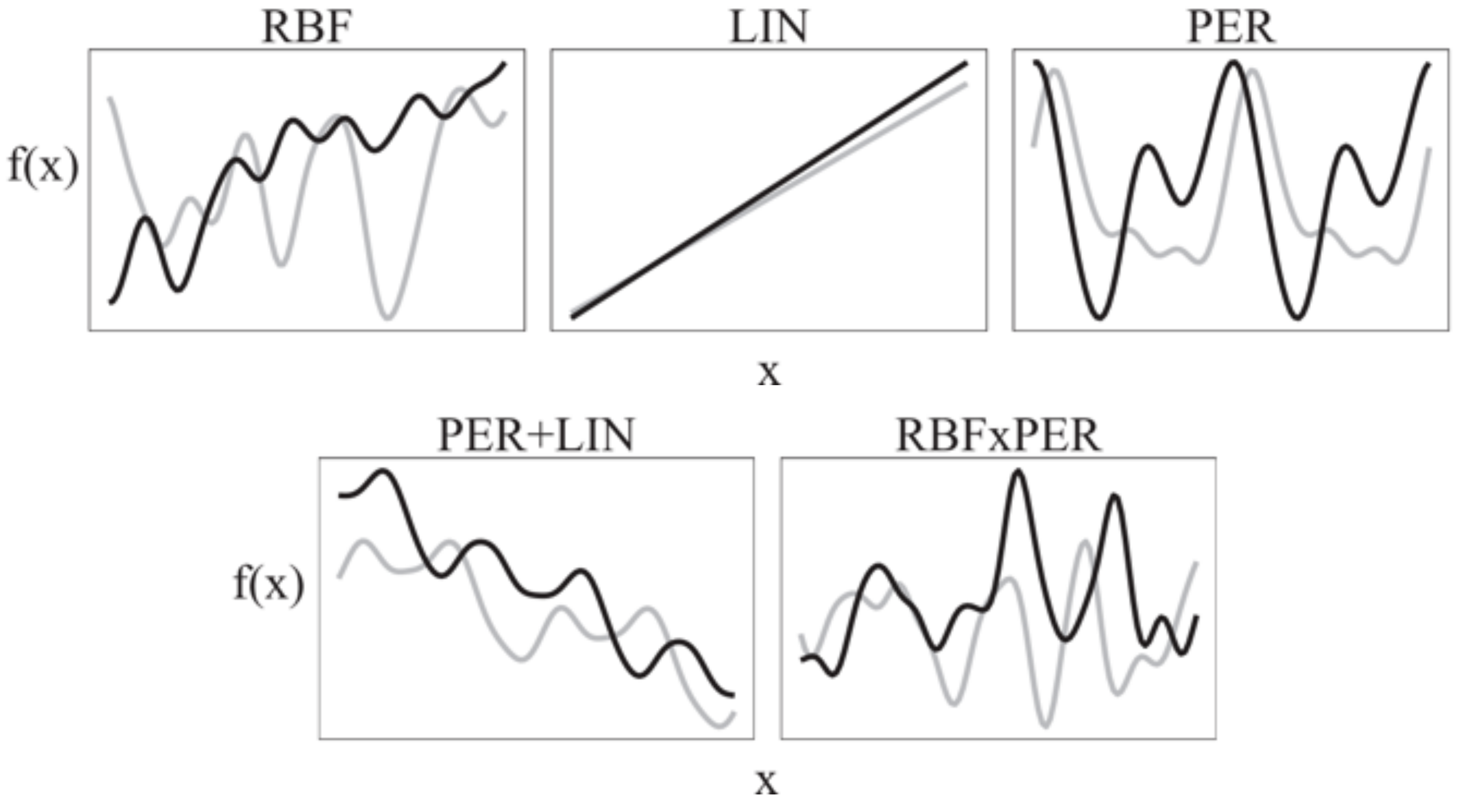
GPs provide the best predictions for human function learning

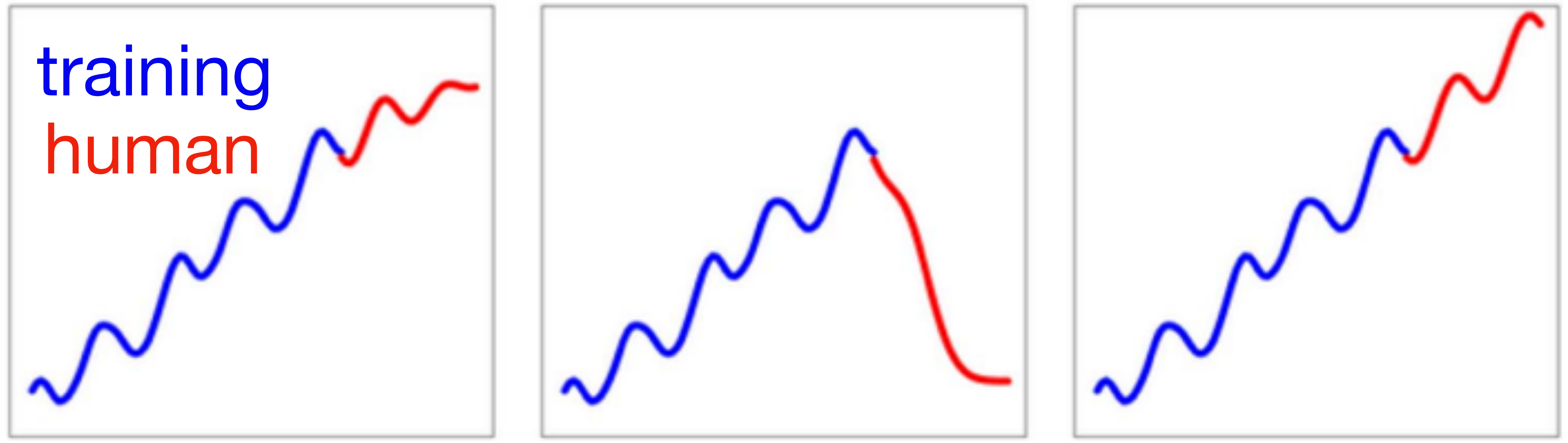
Extrapolation



Griffiths, Lucas, & Williams, (*Neurips* 2008)

Compositional functions

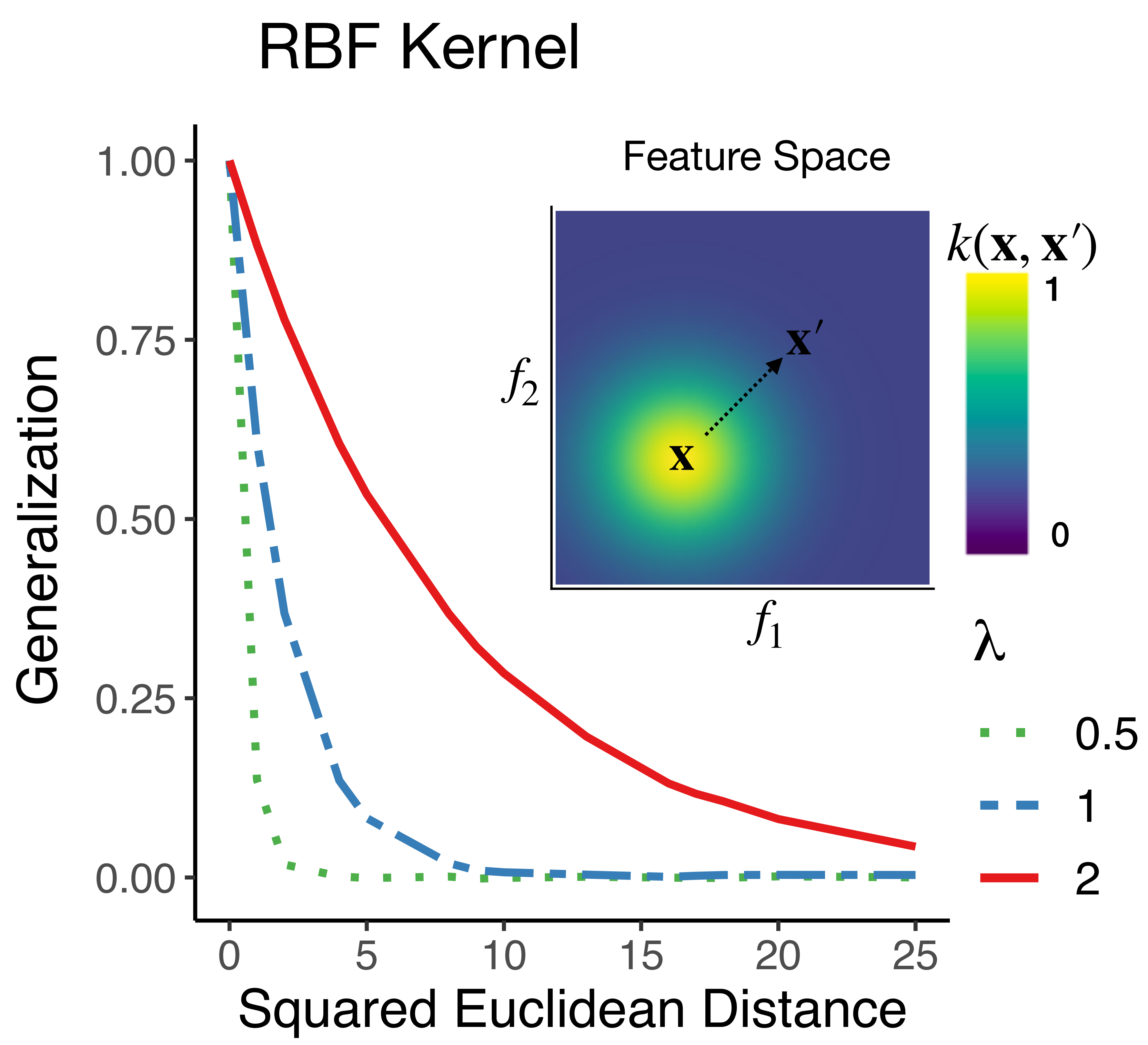




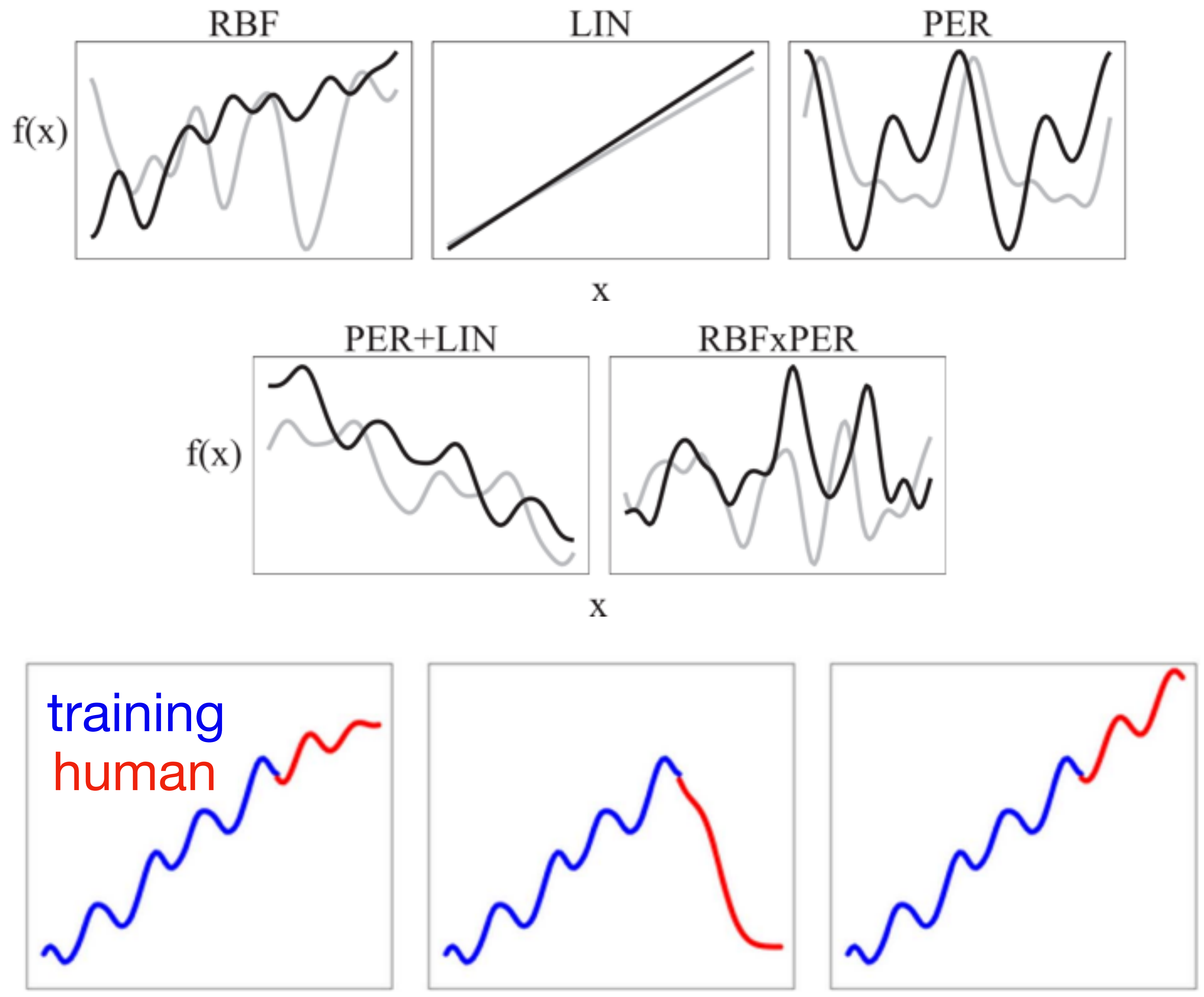
Schulz et al., (*CogPsych* 2017)

Duality of GP function learning

Kernel provides an explicit **similarity** metric



Kernels can be compositionally combined, similar to how we can combine **rules** to create new ones



Wu et al,. (*AnnRevPsych* 2025)

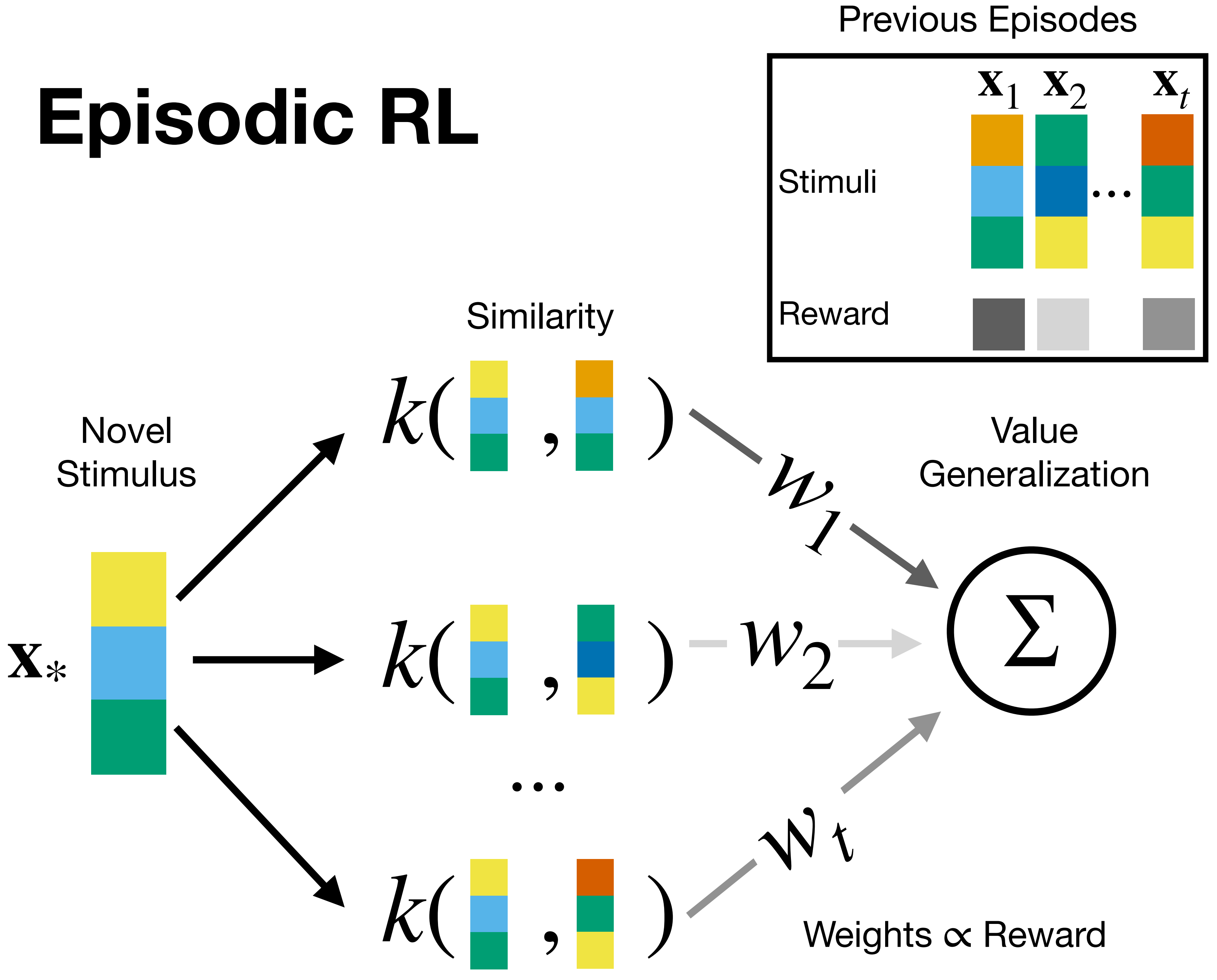
Schulz et al., (*CogPsych* 2017)

Connection to RL

Connection to RL

- Episodic RL for generalization in new settings
  (Gershman & Daw, *AnnRevPsych* 2017; Bottvinick et al., *TICS* 2019)
  - Store a memory of each previously encountered stimuli $\mathbf{x}$ and it's reward y
  - Predict the value of new stimuli based on a similarity-weighted sum of past episodes

Episodic RL



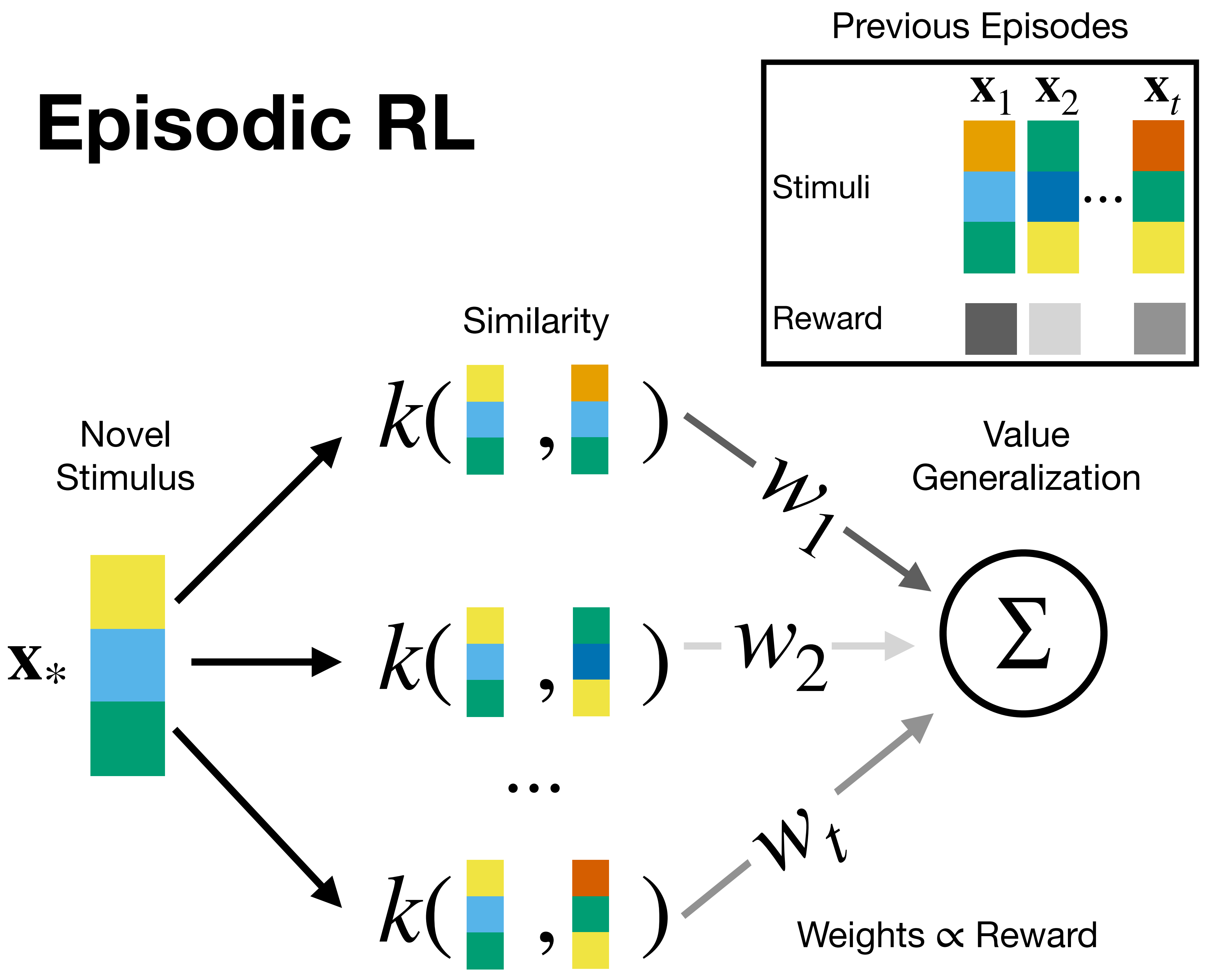
Connection to RL

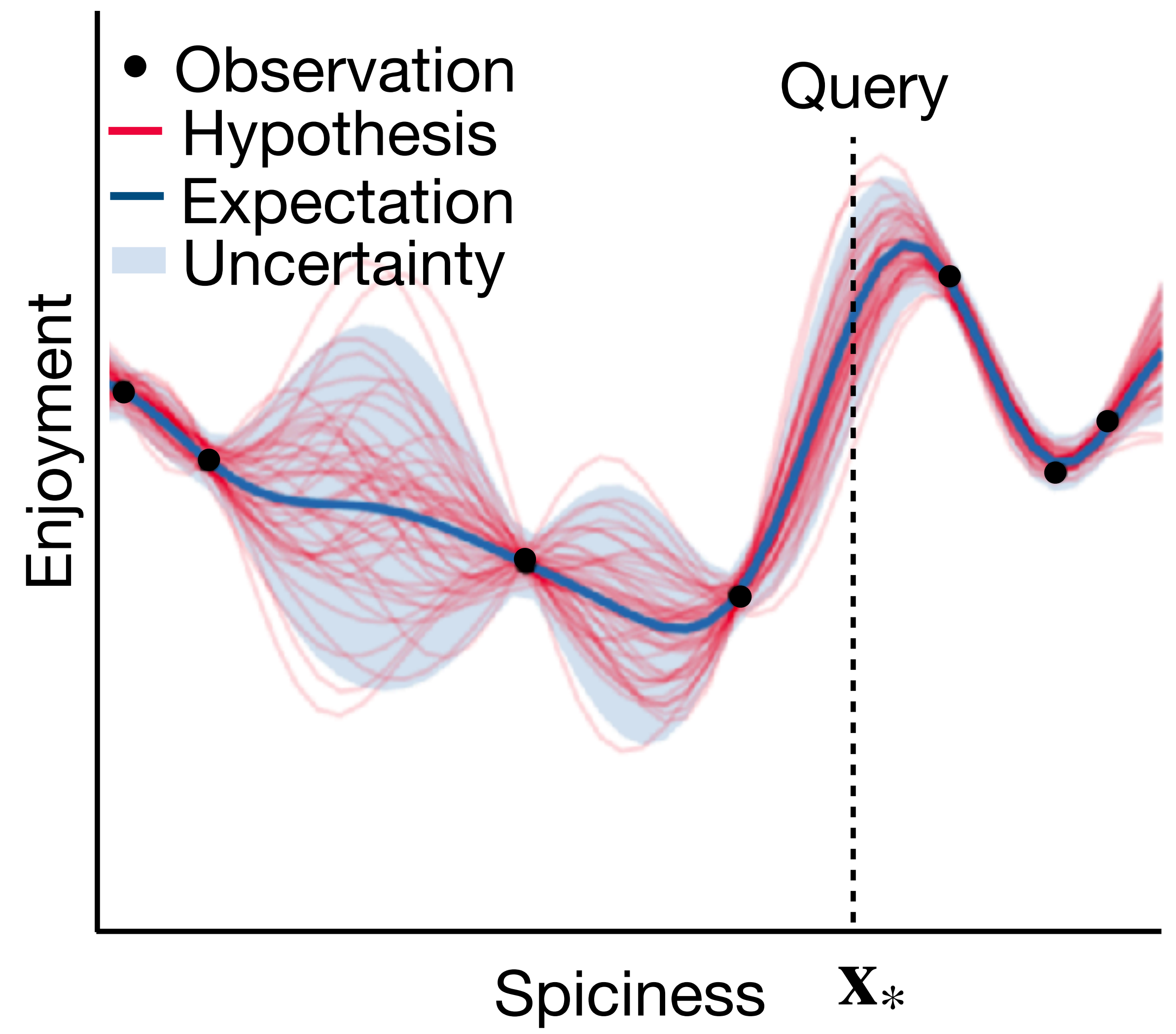
- Episodic RL for generalization in new settings
(Gershman & Daw, *AnnRevPsych* 2017; Bottvinick et al., *TICS* 2019)
  - Store a memory of each previously encountered stimuli $\mathbf{x}$ and it's reward y
  - Predict the value of new stimuli based on a similarity-weighted sum of past episodes
- GPs provide a Bayesian analogue of Episodic RL
  - Using an RBF kernel as the similarity metric, Episodic RL is equivalent to the GP posterior mean
(Poggio & Bizzi, *Nature* 2004; Sutton & Barto, 2018; Jäkel, Schölkopf, & Wichman, *J.MathPsych*, 2008)
  - Yet GPs provide uncertainty estimates, which is essential for defining which states to explore!

$$m(\mathbf{x}_* \,|\, \mathscr{D}) = \mathbf{k}_*^\top (\mathbf{K} + \sigma^2 \mathbf{I})^{-1} \mathbf{y}$$

$$v(\mathbf{x}_* \,|\, \mathscr{D}) = k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}_*^\top (\mathbf{K} + \sigma^2 \mathbf{I})^{-1} \mathbf{k}_*$$

Episodic RL

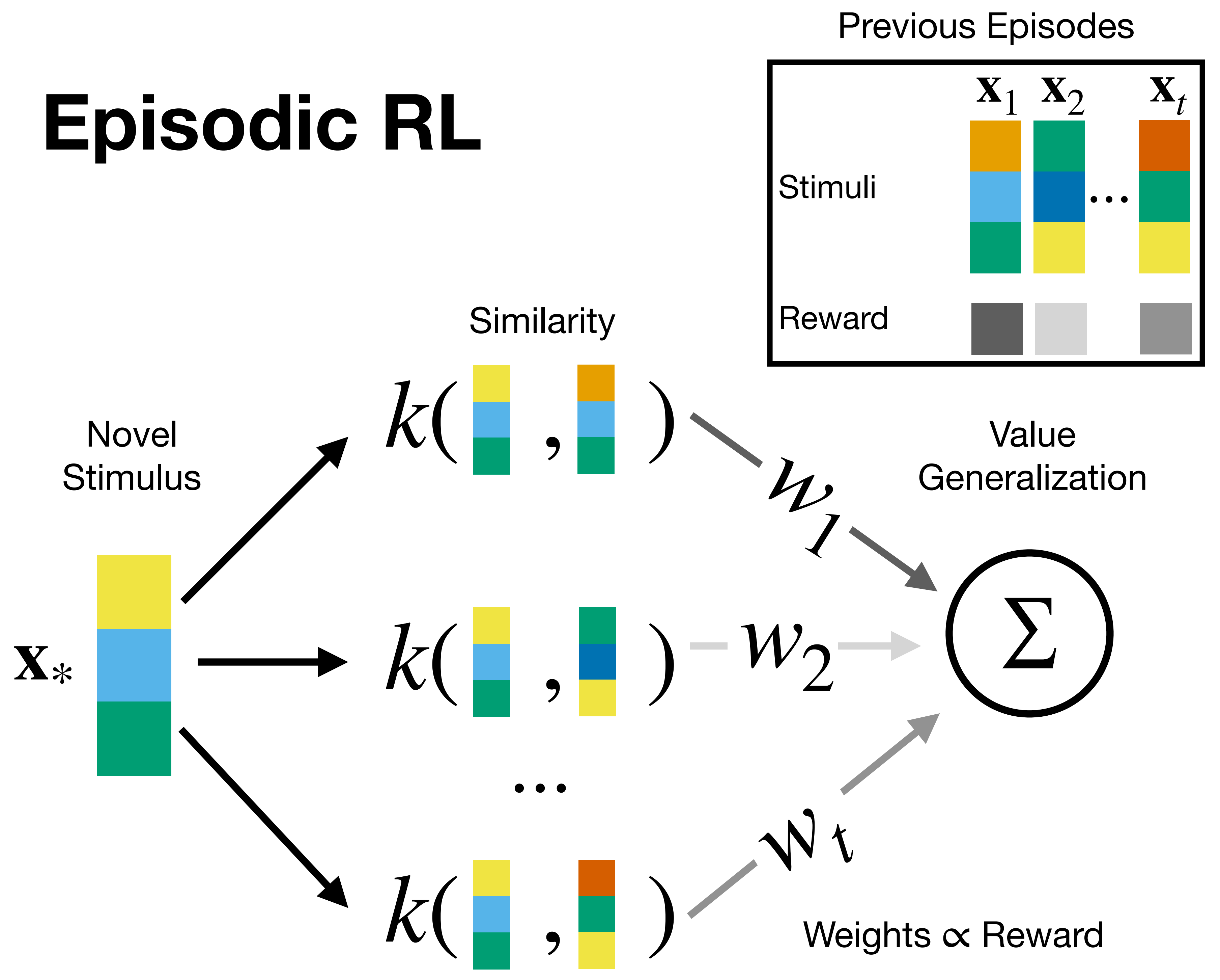




Connection to RL

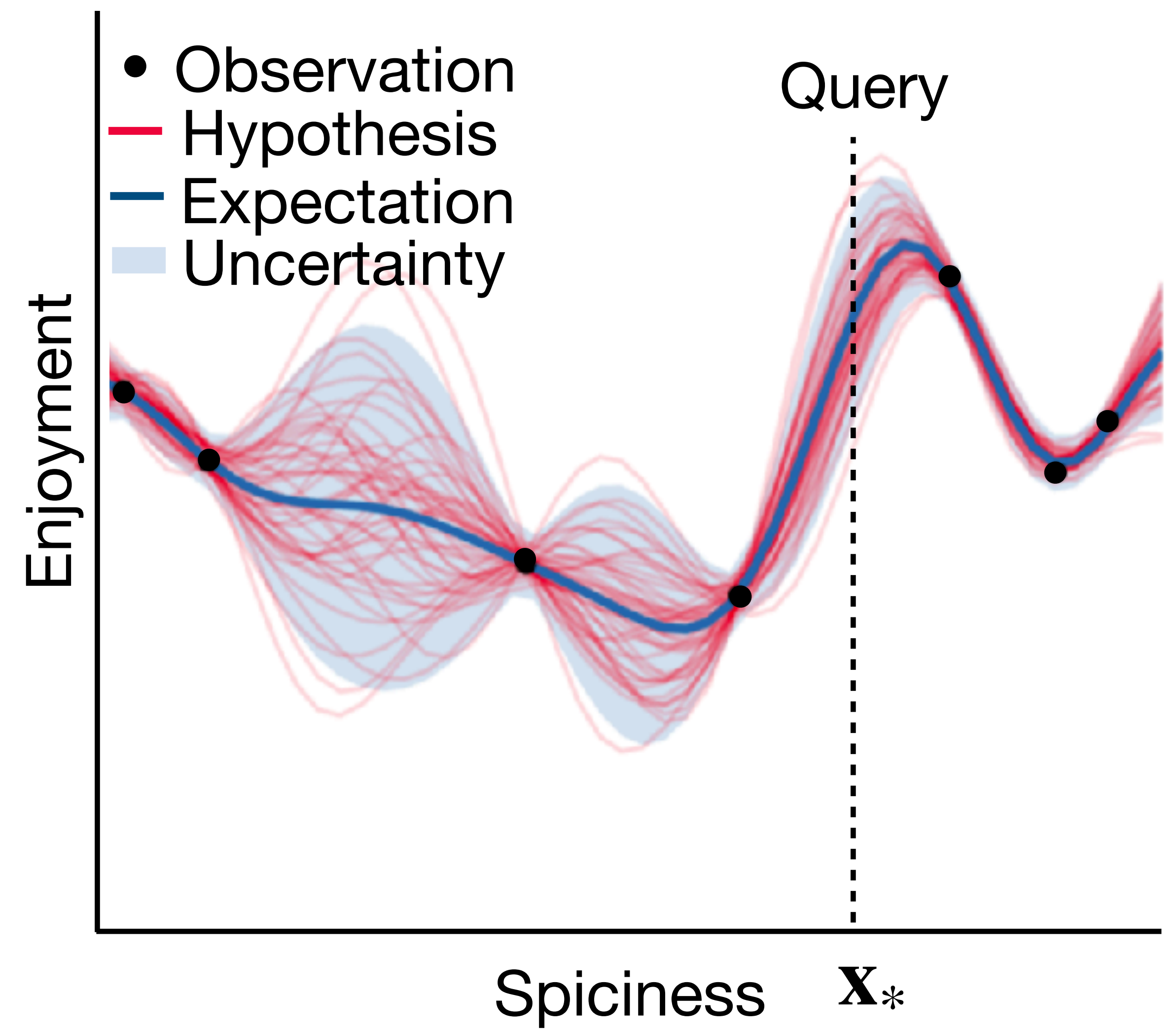
- Episodic RL for generalization in new settings
(Gershman & Daw, *AnnRevPsych* 2017; Bottvinick et al., *TICS* 2019)
    - Store a memory of each previously encountered stimuli $\mathbf{x}$ and it's reward y
    - Predict the value of new stimuli based on a similarity-weighted sum of past episodes
- GPs provide a Bayesian analogue of Episodic RL
    - Using an RBF kernel as the similarity metric, Episodic RL is equivalent to the GP posterior mean
    (Poggio & Bizzi, *Nature* 2004; Sutton & Barto, 2018; Jäkel, Schölkopf, & Wichman, *J.MathPsych*, 2008)
    - Yet GPs provide uncertainty estimates, which is essential for defining which states to explore!

Episodic RL



$$m(\mathbf{x}_* \mid \mathscr{D}) = \mathbf{k}_*^\top(\mathbf{K} + \sigma^2\mathbf{I})^{-1}\mathbf{y} = \sum_{i=1}^{N} w_i k(\mathbf{x}, \mathbf{x}') \quad \text{where} \quad \mathbf{w} = [\mathbf{K} + \sigma^2\mathbf{I}]^{-1}\mathbf{y}$$

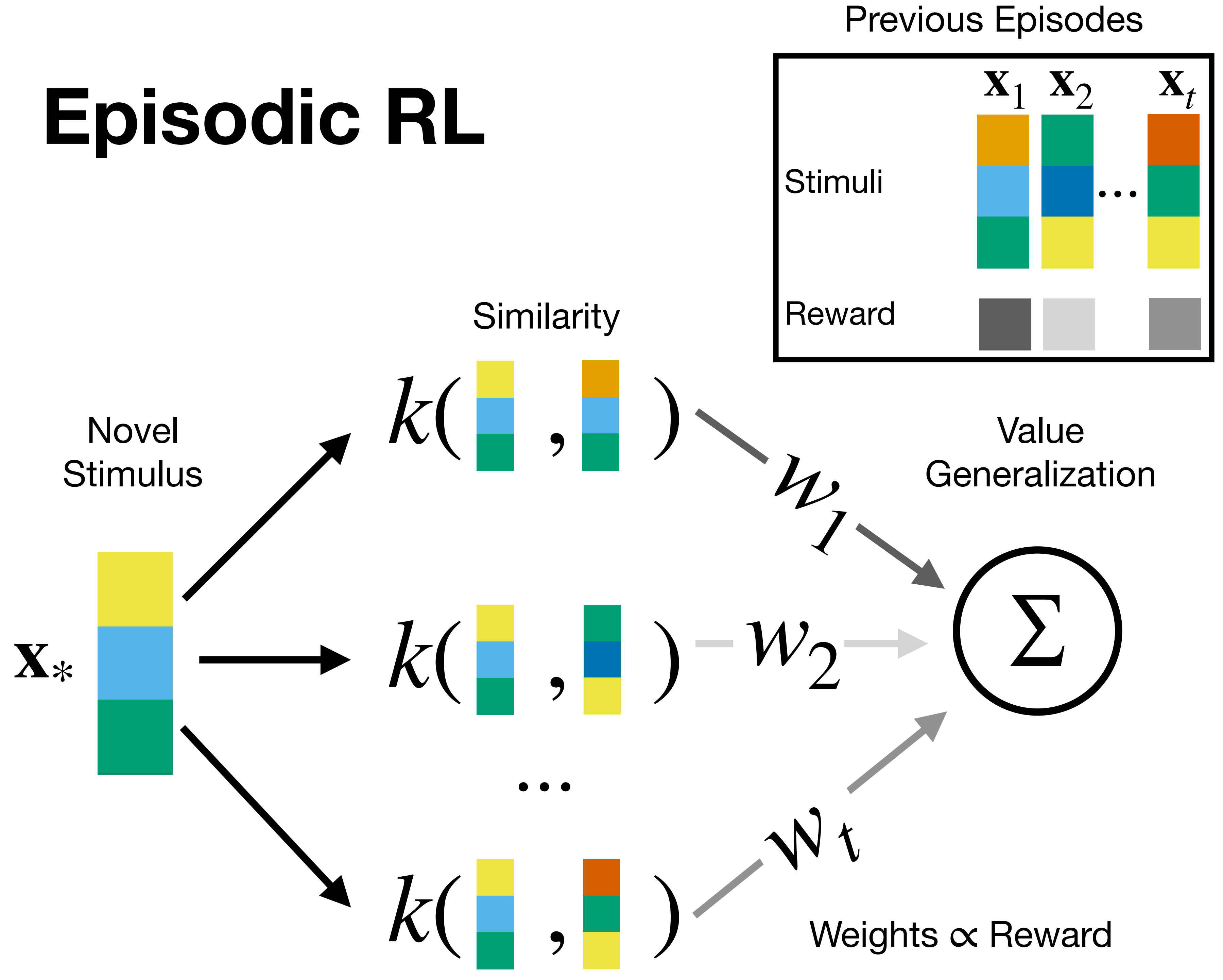
$$v(\mathbf{x}_* \mid \mathscr{D}) = k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}_*^\top(\mathbf{K} + \sigma^2\mathbf{I})^{-1}\mathbf{k}_*$$

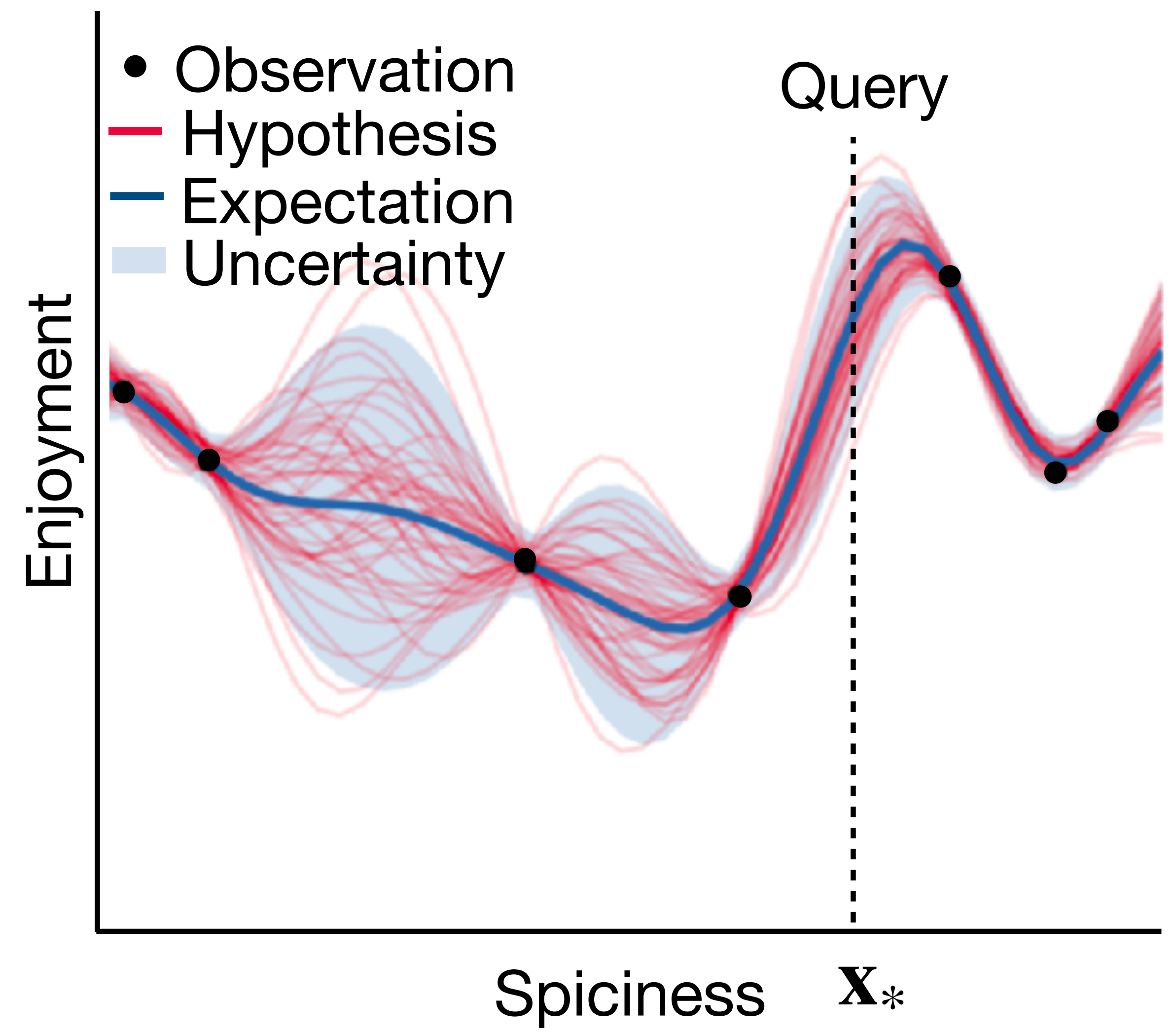


Connection to RL

- Episodic RL for generalization in new settings
  (Gershman & Daw, *AnnRevPsych* 2017; Bottvinick et al., *TICS* 2019)
  - Store a memory of each previously encountered stimuli $\mathbf{x}$ and it's reward y
  - Predict the value of new stimuli based on a similarity-weighted sum of past episodes
- GPs provide a Bayesian analogue of Episodic RL
  - Using an RBF kernel as the similarity metric, Episodic RL is equivalent to the GP posterior mean
    (Poggio & Bizzi, *Nature* 2004; Sutton & Barto, 2018; Jäkel, Schölkopf, & Wichman, *J.MathPsych*, 2008)
  - Yet GPs provide uncertainty estimates, which is essential for defining which states to explore!

Episodic RL





$$m(\mathbf{x}_* \,|\, \mathscr{D}) = \mathbf{k}_*^\top (\mathbf{K} + \sigma^2 \mathbf{I})^{-1} \mathbf{y} \;=\; \sum_{i=1}^{N} w_i k(\mathbf{x}, \mathbf{x}') \quad \text{where} \quad \mathbf{w} = [\mathbf{K} + \sigma^2 \mathbf{I}]^{-1} \mathbf{y}$$

weights — similarity

$$v(\mathbf{x}_* \,|\, \mathscr{D}) = k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}_*^\top (\mathbf{K} + \sigma^2 \mathbf{I})^{-1} \mathbf{k}_*$$

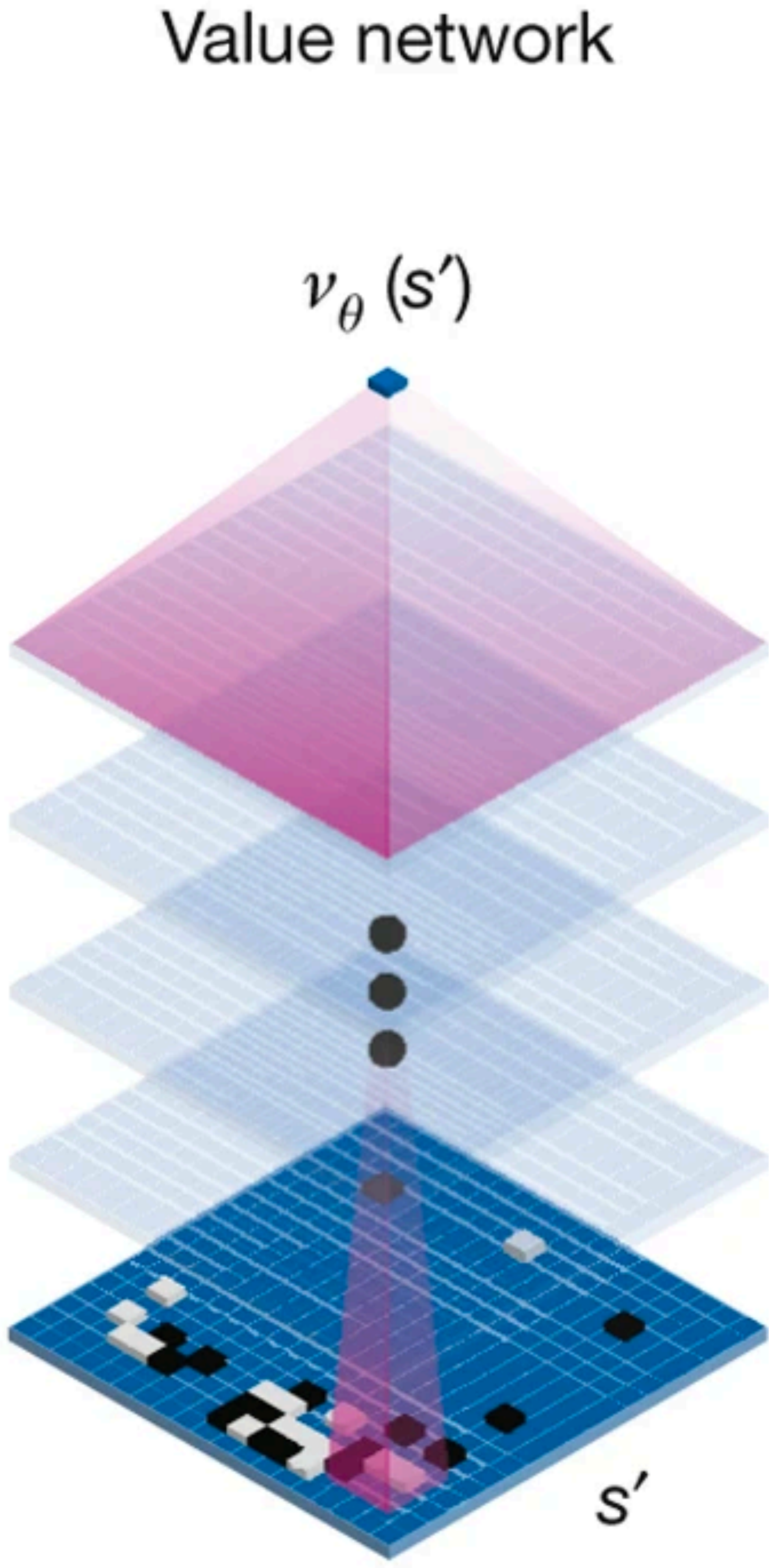
Value function approximation in RL

- Classic function learning is typically a supervised learning problem
  - Given stimulus $\mathbf{x}_*$ predict $f(\mathbf{x}_*)$
- Value function approximation is a key method for generalization in RL
  - Use function learning mechanisms for inferring *implicit* value of novel states: $V(s') = f(s')$
  - Implement a policy on the basis of value: $\pi(s') \propto \exp(V(\mathbf{s}'))$
- AlphaGo uses a deep neural network for value function approximation
  - DNNs are simply a universal function approximator (Cybenko, 1989)
  - But for understanding human behavior, GPs offer better interpretability due to psychologically meaningful parameters
  - **GPs are equivalent to an infinitely wide deep neural network** (Neal, 1996)
- After the break, I will present some of my research using GPs to model human generalization in RL

Silver et al., (*Nature* 2016)

Value function approximation in RL

- Classic function learning is typically a supervised learning problem
  - Given stimulus $\mathbf{x}_*$ predict $f(\mathbf{x}_*)$
- Value function approximation is a key method for generalization in RL
  - Use function learning mechanisms for inferring *implicit* value of novel states: $V(s') = f(s')$
  - Implement a policy on the basis of value: $\pi(s') \propto \exp(V(\mathbf{s}'))$
- AlphaGo uses a deep neural network for value function approximation
  - DNNs are simply a universal function approximator (Cybenko, 1989)
  - But for understanding human behavior, GPs offer better interpretability due to psychologically meaningful parameters
  - **GPs are equivalent to an infinitely wide deep neural network** (Neal, 1996)
- After the break, I will present some of my research using GPs to model human generalization in RL



Silver et al., (*Nature* 2016)

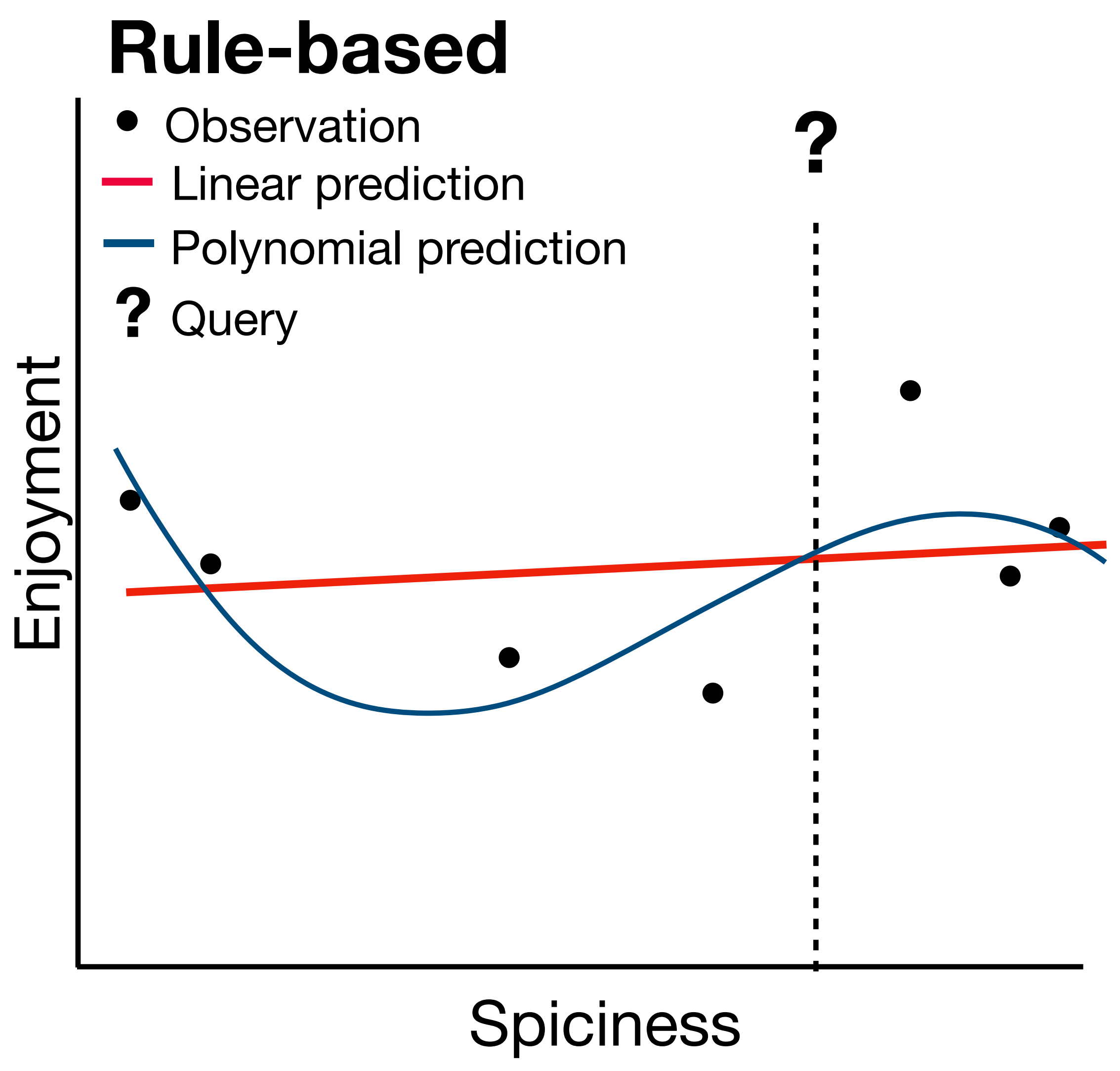
Interim summary

- Function learning is a regression problem
- Early **rule-based theories** assumed humans learn functions by picking specific class of functions and then optimizing the weights (as in linear or parametric regression) —> Brittle and lacked flexibility
- **Similarity-based methods** used ANNs to encode the generic principle that similar inputs produce similar outputs —> failed to capture systematic biases in how humans extrapolate
- **Hybrid approaches** using GP regression offer a Bayesian framework, combining kernel similarity and rule-like compositionality of kernels

Regression task

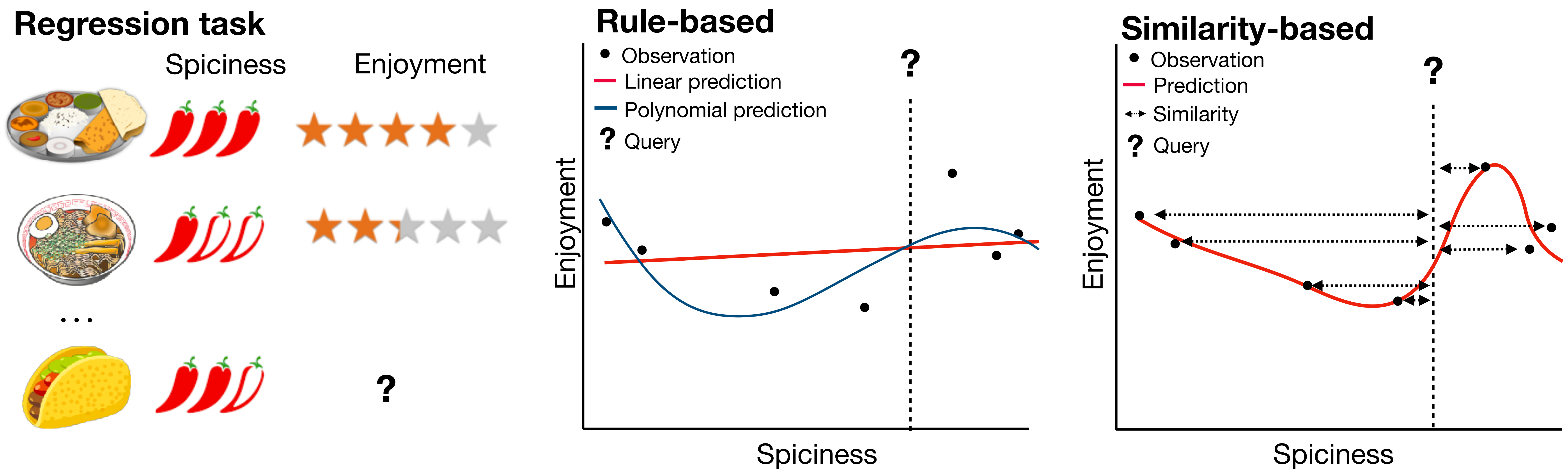
Interim summary

- Function learning is a regression problem
- Early **rule-based theories** assumed humans learn functions by picking specific class of functions and then optimizing the weights (as in linear or parametric regression) —> Brittle and lacked flexibility
- **Similarity-based methods** used ANNs to encode the generic principle that similar inputs produce similar outputs —> failed to capture systematic biases in how humans extrapolate
- **Hybrid approaches** using GP regression offer a Bayesian framework, combining kernel similarity and rule-like compositionality of kernels



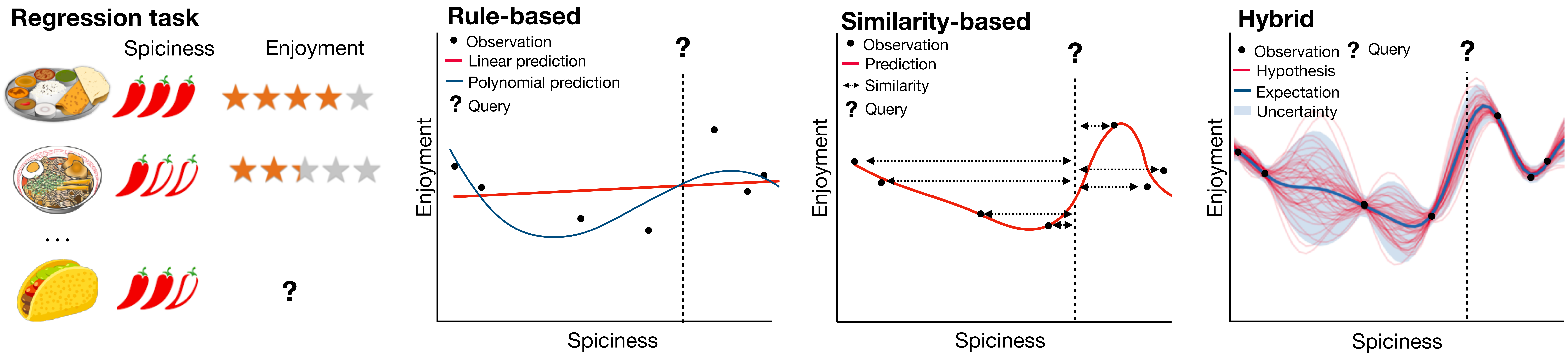
Interim summary

- Function learning is a regression problem
- Early **rule-based theories** assumed humans learn functions by picking specific class of functions and then optimizing the weights (as in linear or parametric regression) —> Brittle and lacked flexibility
- **Similarity-based methods** used ANNs to encode the generic principle that similar inputs produce similar outputs —> failed to capture systematic biases in how humans extrapolate
- **Hybrid approaches** using GP regression offer a Bayesian framework, combining kernel similarity and rule-like compositionality of kernels



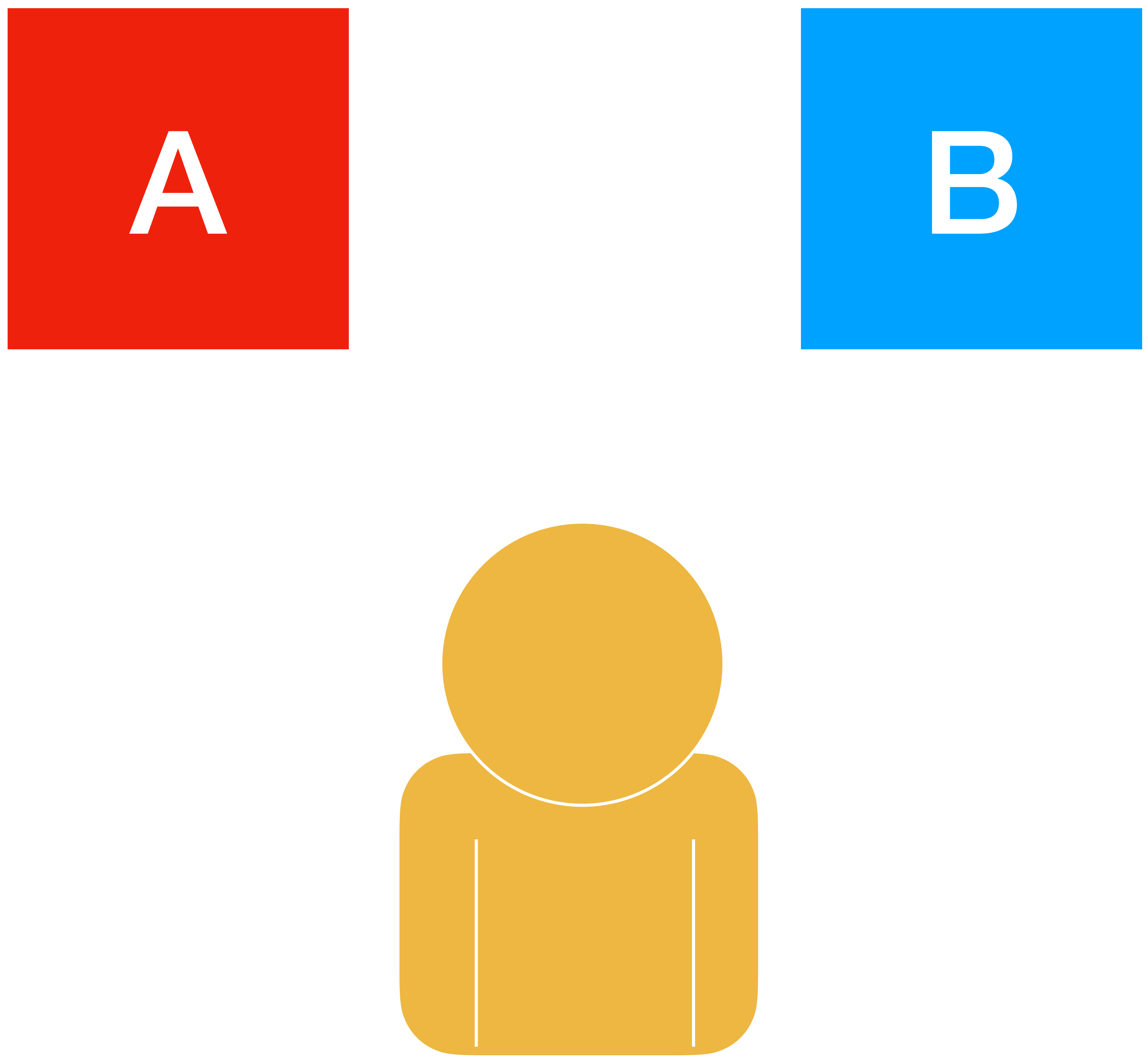
Interim summary

- Function learning is a regression problem
- Early **rule-based theories** assumed humans learn functions by picking specific class of functions and then optimizing the weights (as in linear or parametric regression) —> Brittle and lacked flexibility
- **Similarity-based methods** used ANNs to encode the generic principle that similar inputs produce similar outputs —> failed to capture systematic biases in how humans extrapolate
- **Hybrid approaches** using GP regression offer a Bayesian framework, combining kernel similarity and rule-like compositionality of kernels

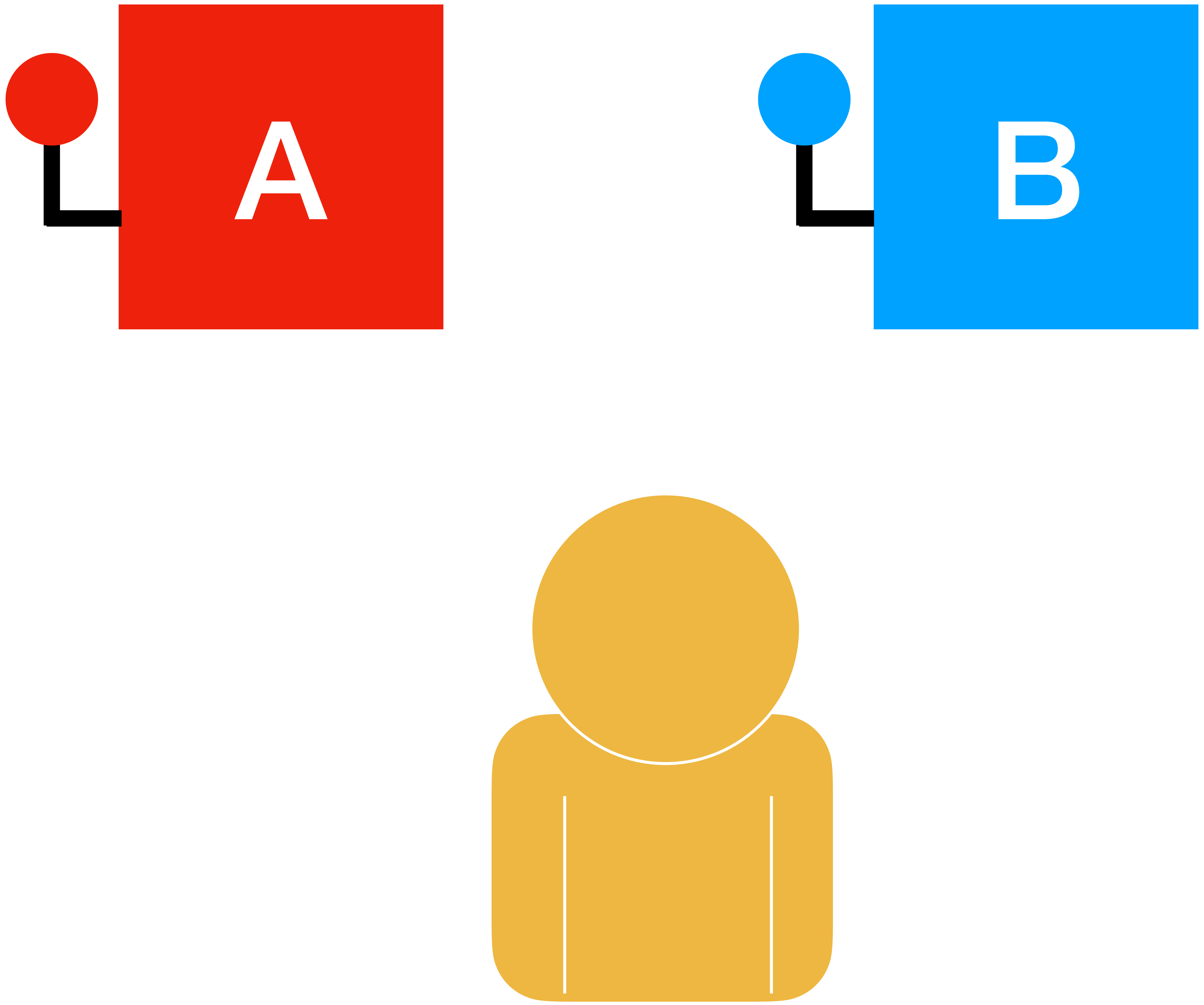


5 minute break

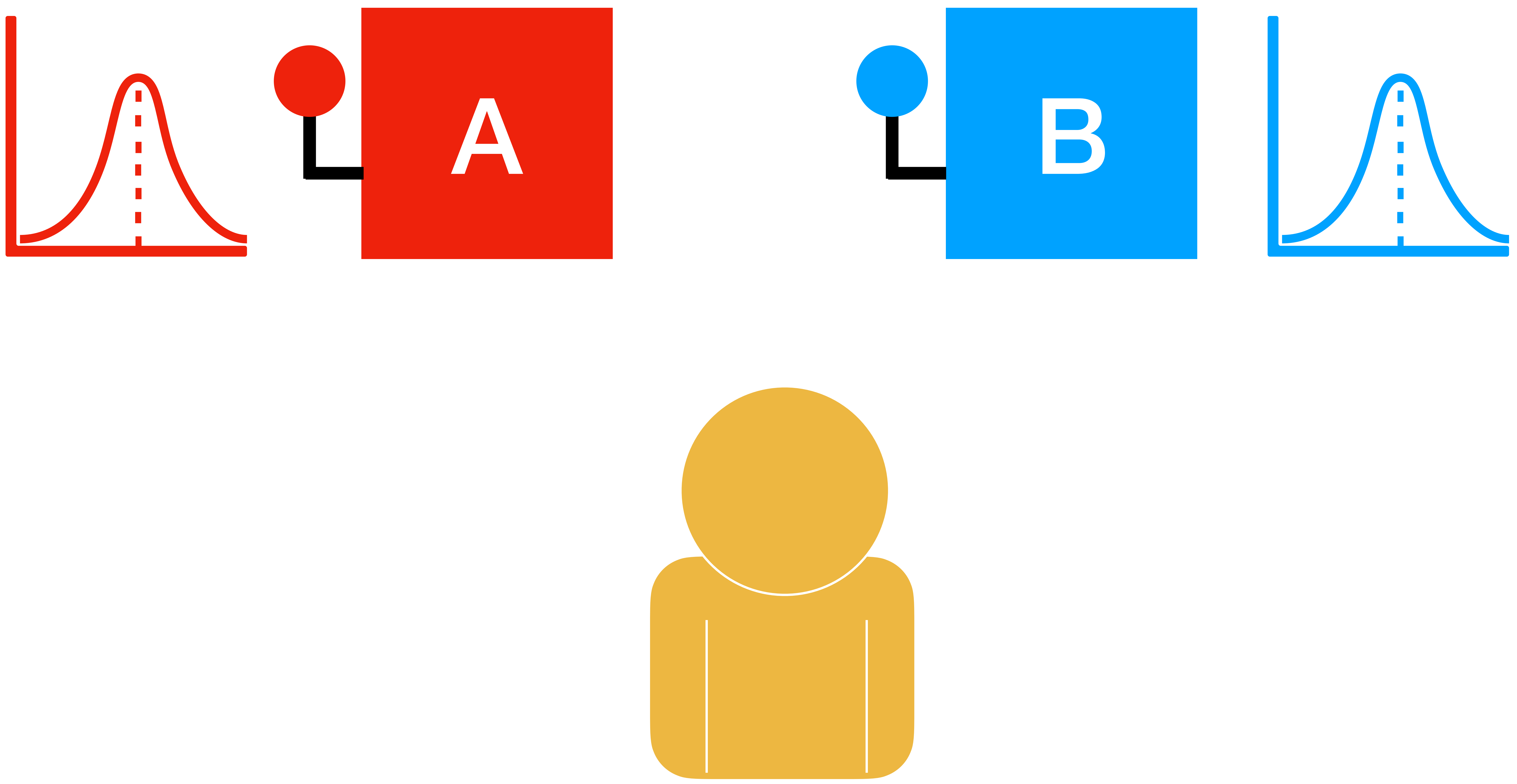
Human learning in the lab



Human learning in the lab

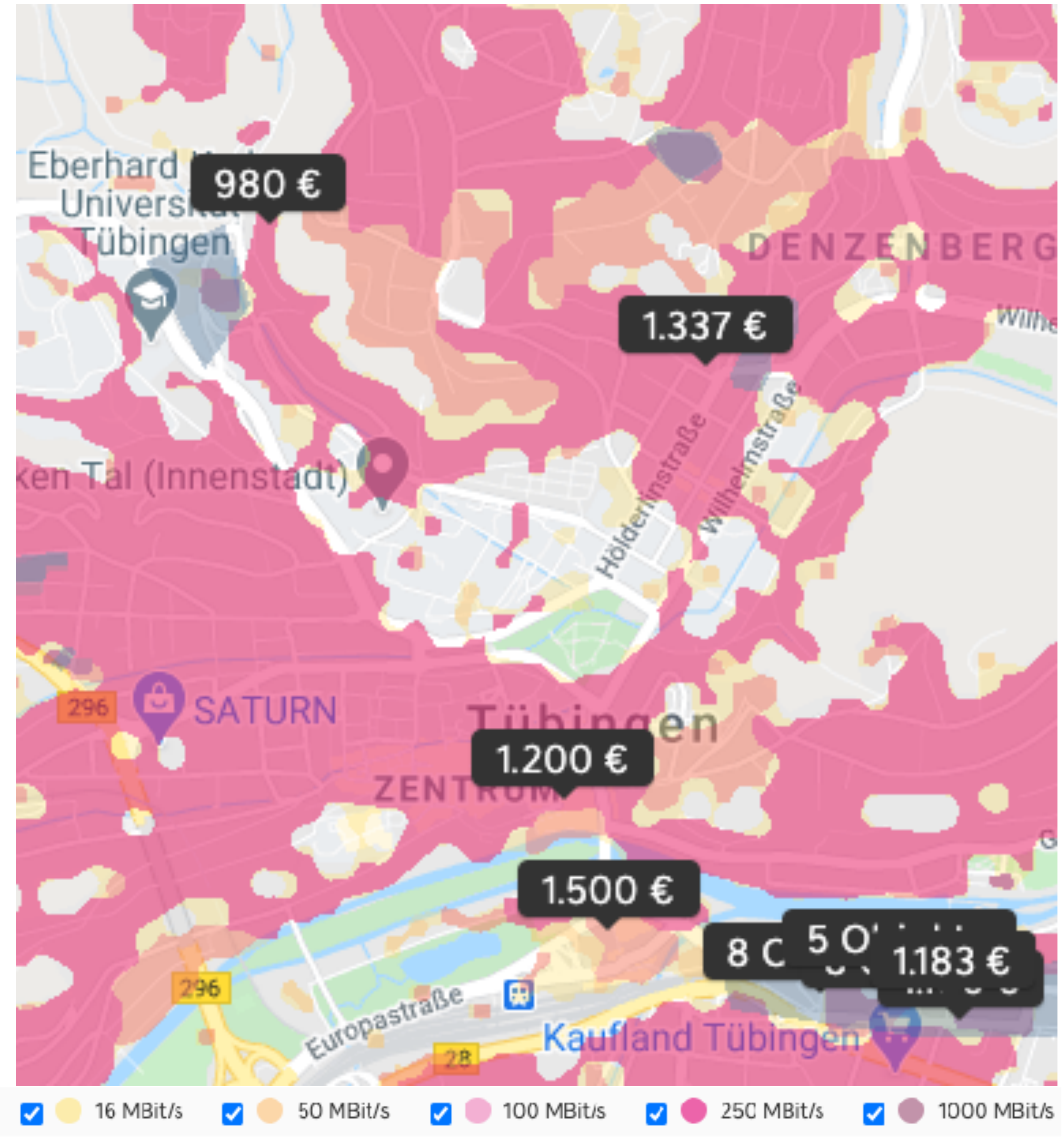


Human learning in the lab



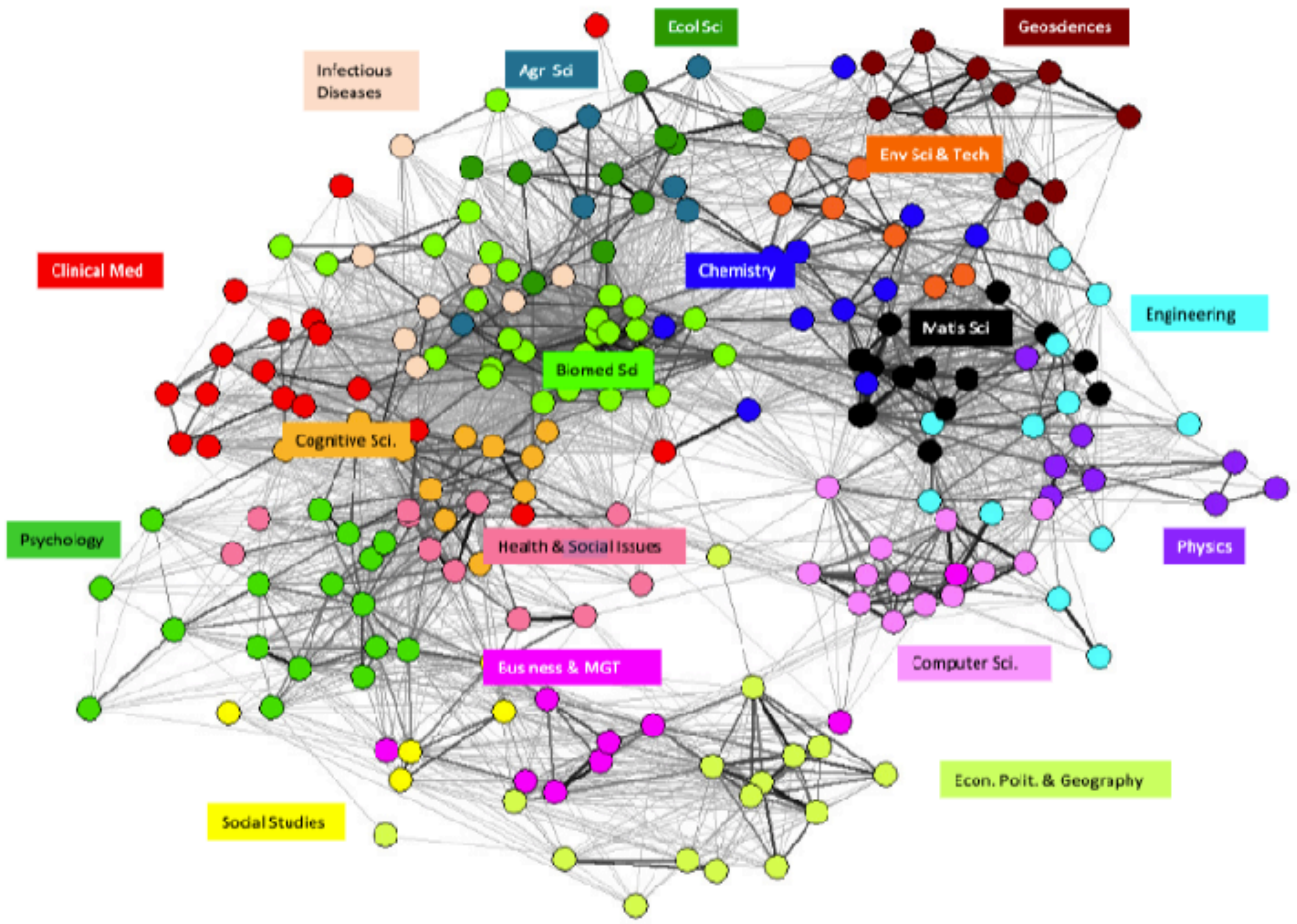
Real life problems

Finding a place to live

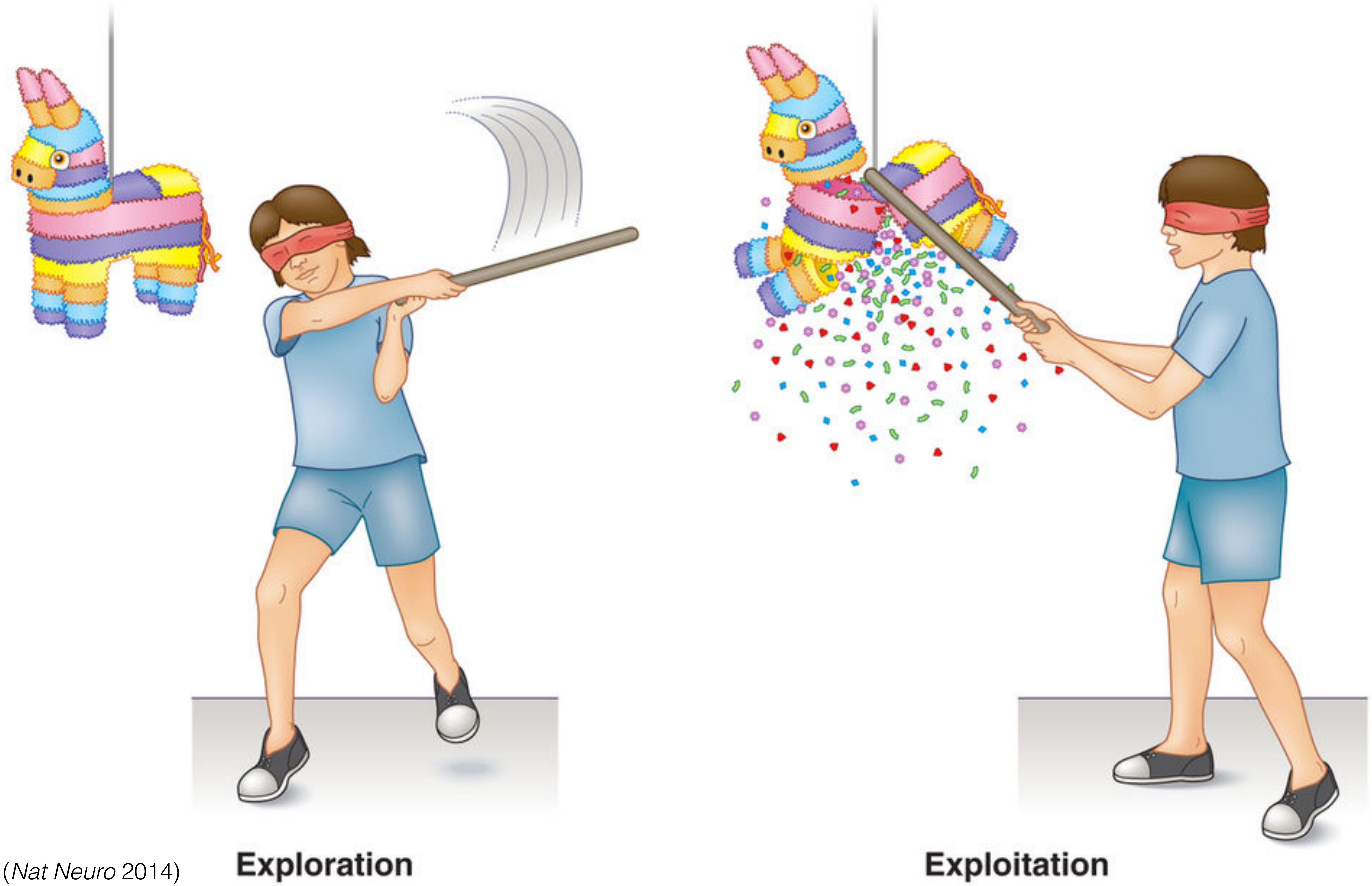


Picking what to eat

Choosing a research topic



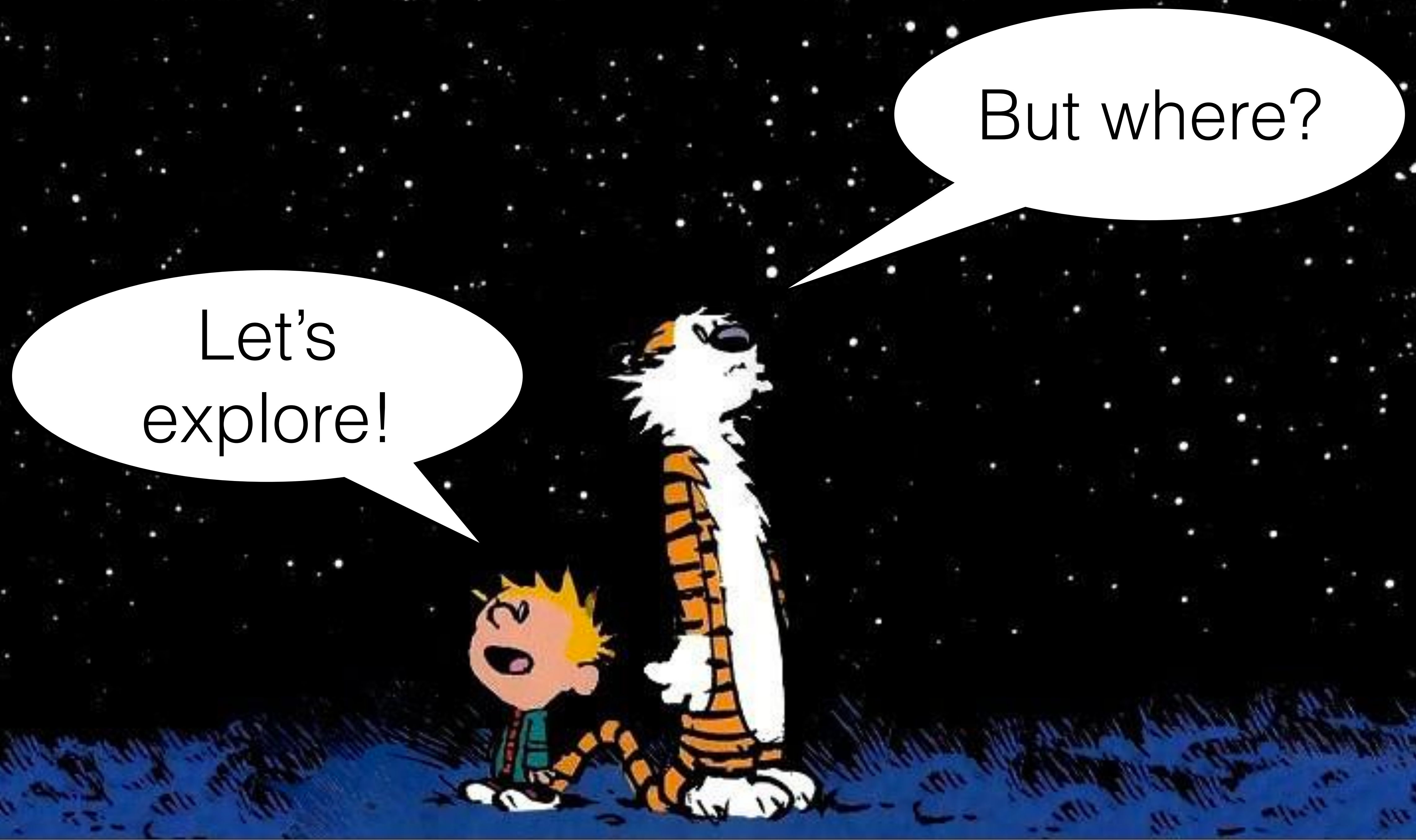
Exploration-Exploitation Dilemma



Herzfeld & Shadmehr (*Nat Neuro* 2014)

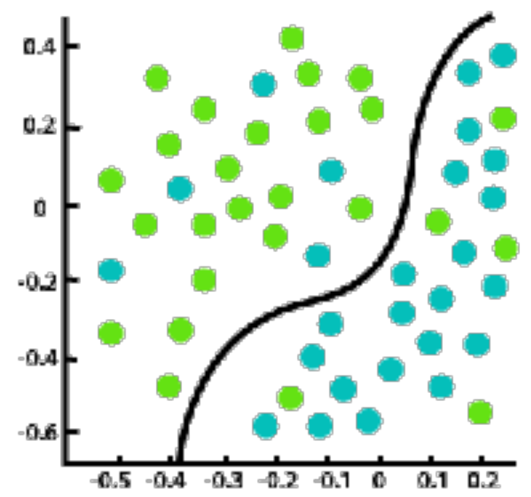

But where?
Let's explore!

How do people navigate vast environments when we cannot explore all possibilities?

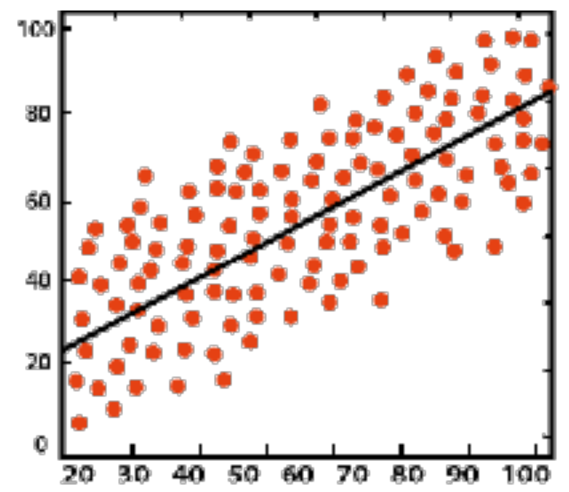


Generalization in RL

Classification

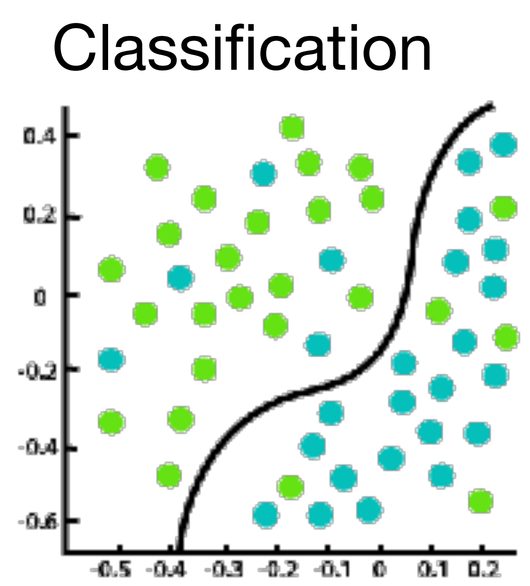


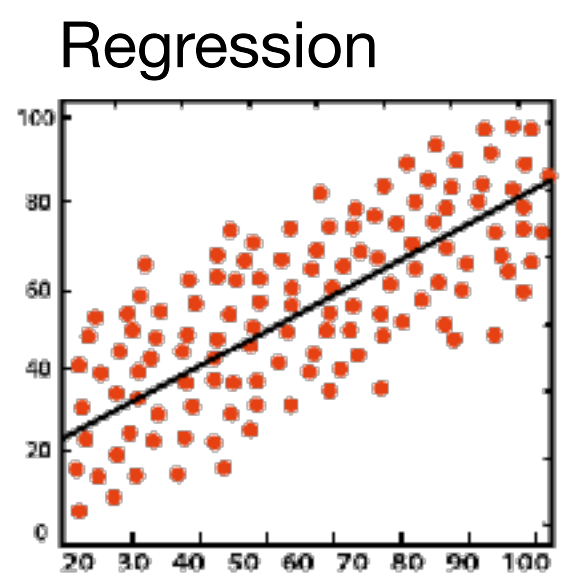
Regression



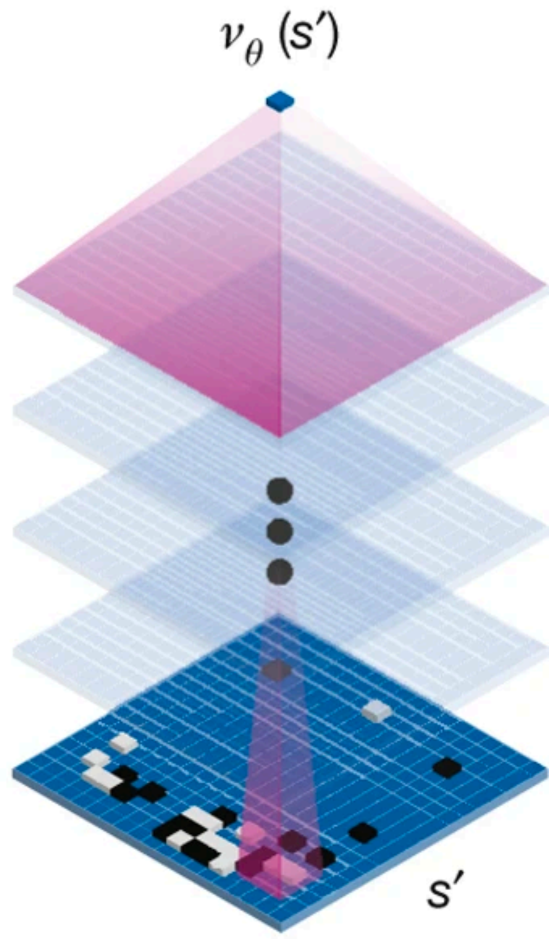
- Shepard formalized generalization as *classification*
- In RL, we can formalize generalization as *regression*: learning a value function

Generalization in RL

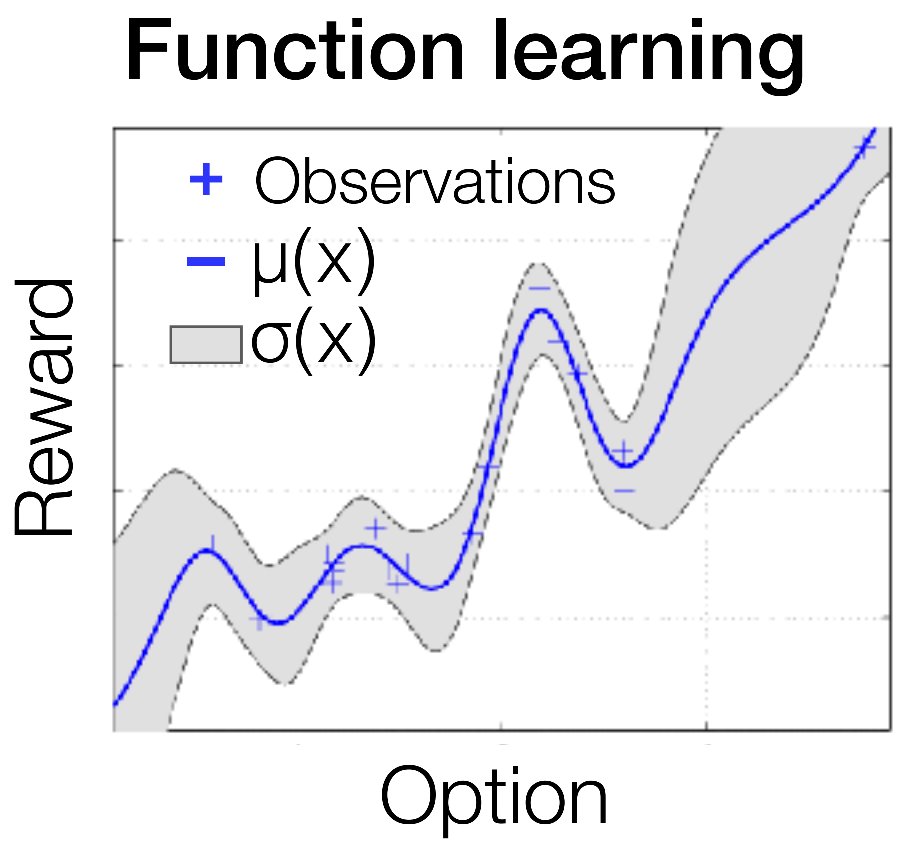




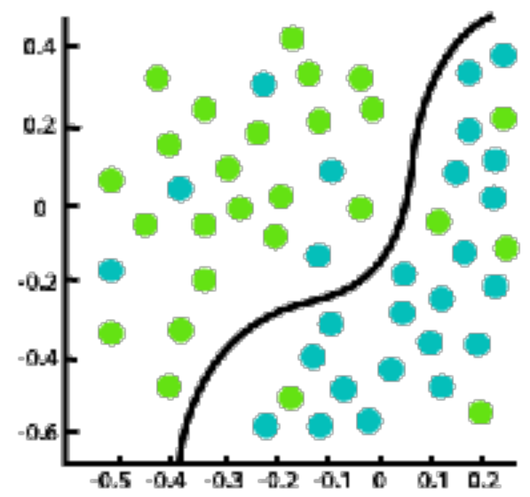
- Shepard formalized generalization as *classification*
- In RL, we can formalize generalization as *regression*: learning a value function
- **Function learning:**
  - Learn an implicit value function mapping states to reward expectations; ubiquitous in modern RL
  - Predict *where* to explore through interpolation and extrapolation

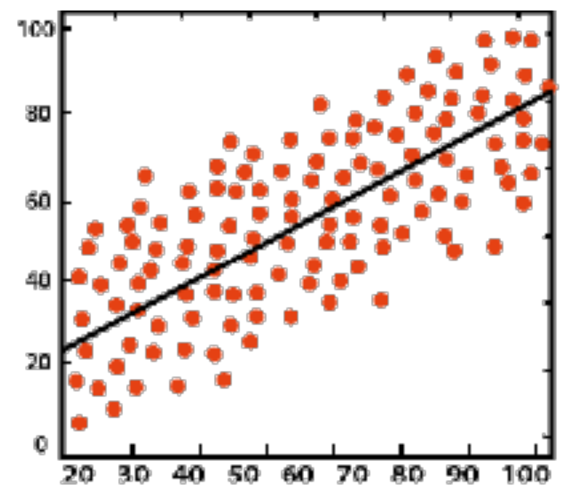


Silver et al., (*Nature* 2016)

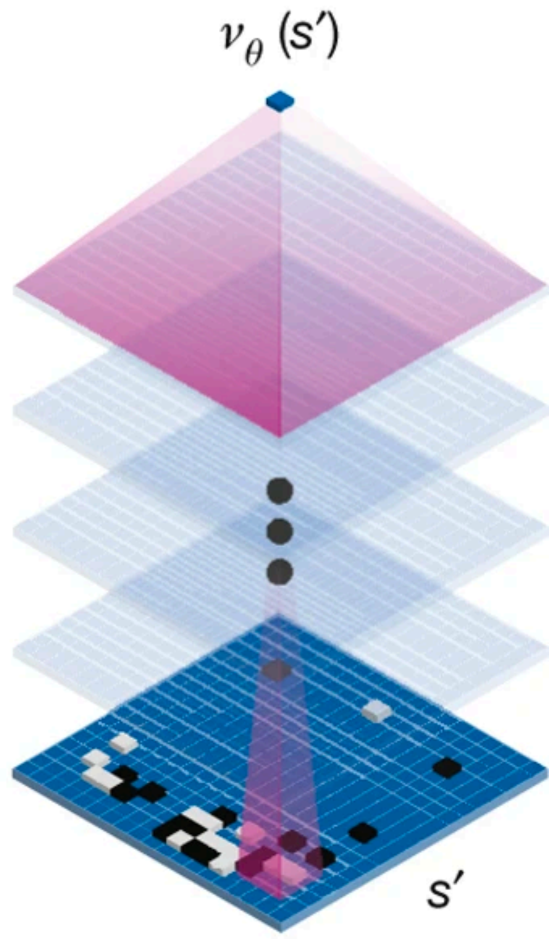


Generalization in RL

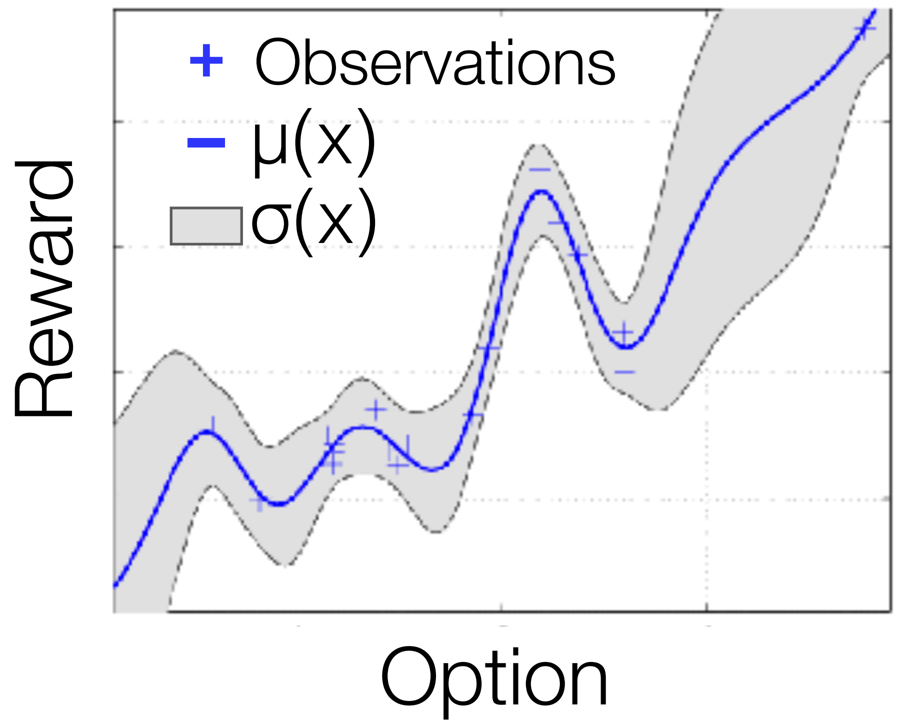




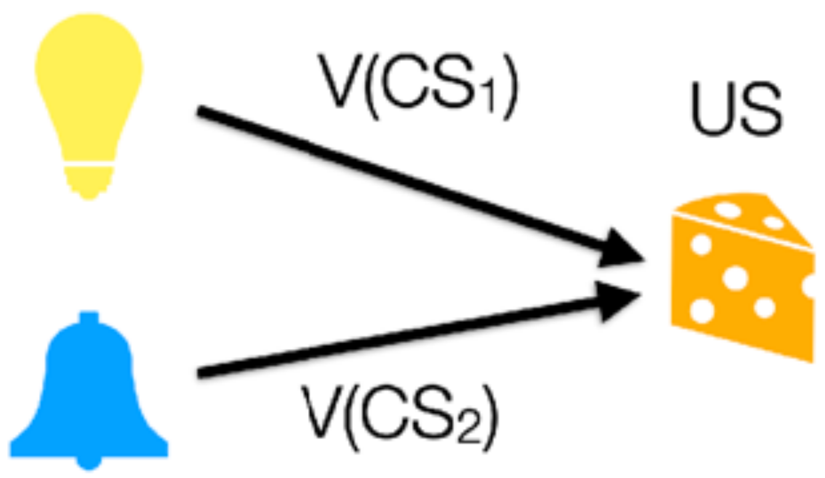
- Shepard formalized generalization as *classification*
- In RL, we can formalize generalization as *regression*: learning a value function
- **Function learning:**
  - Learn an implicit value function mapping states to reward expectations; ubiquitous in modern RL
  - Predict *where* to explore through interpolation and extrapolation

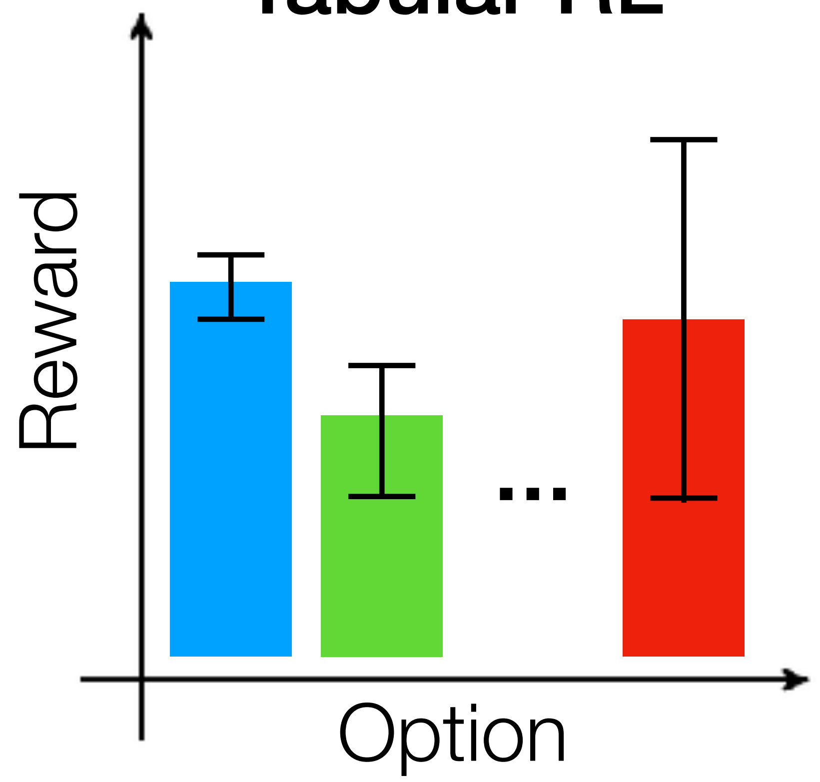


Silver et al., (*Nature* 2016)

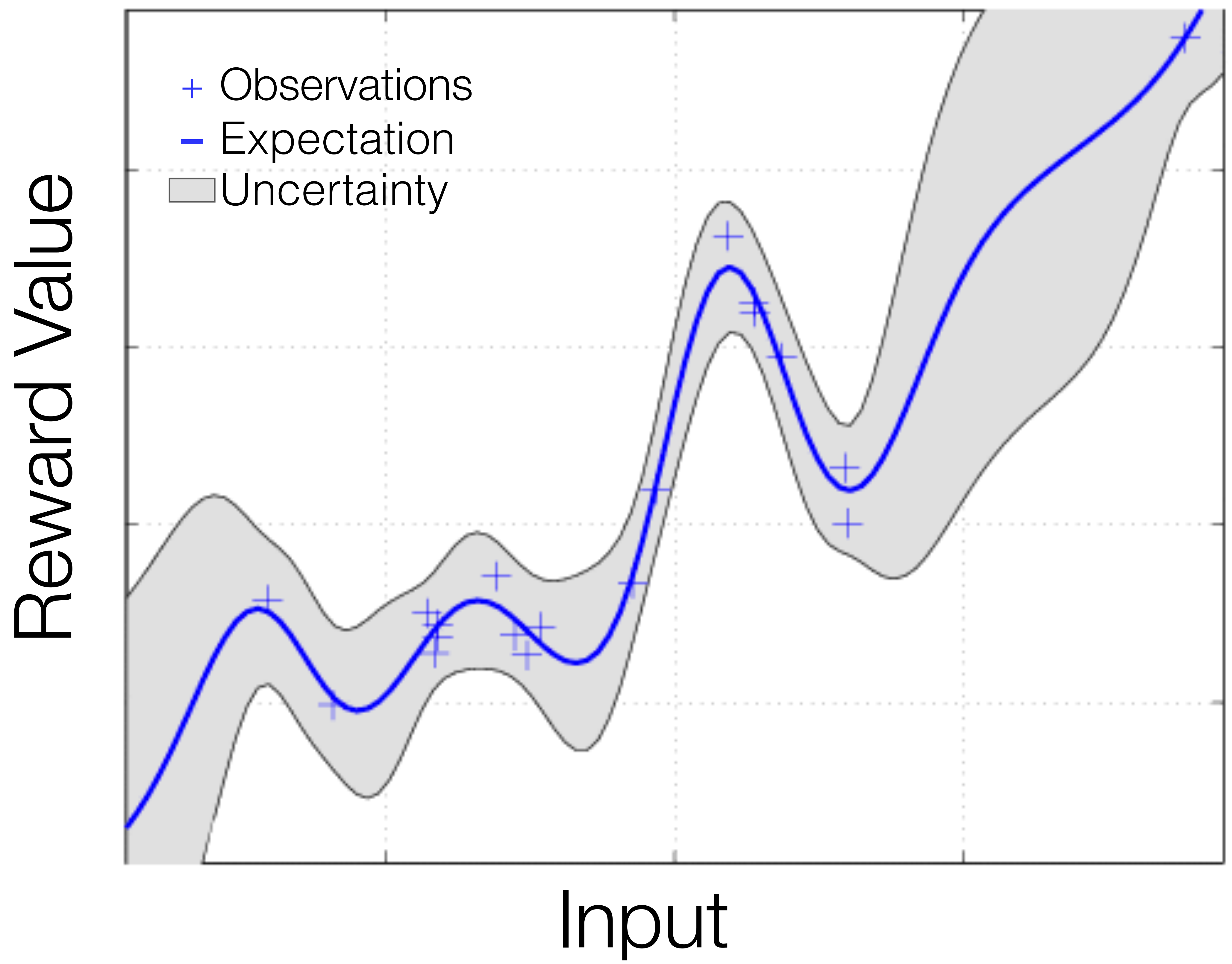


- **Tabular learning:**
  - Traditional associative learning models learn the value of each option independently
  - No guidance about *where to explore,* with novel options defaulting to some prior expectation

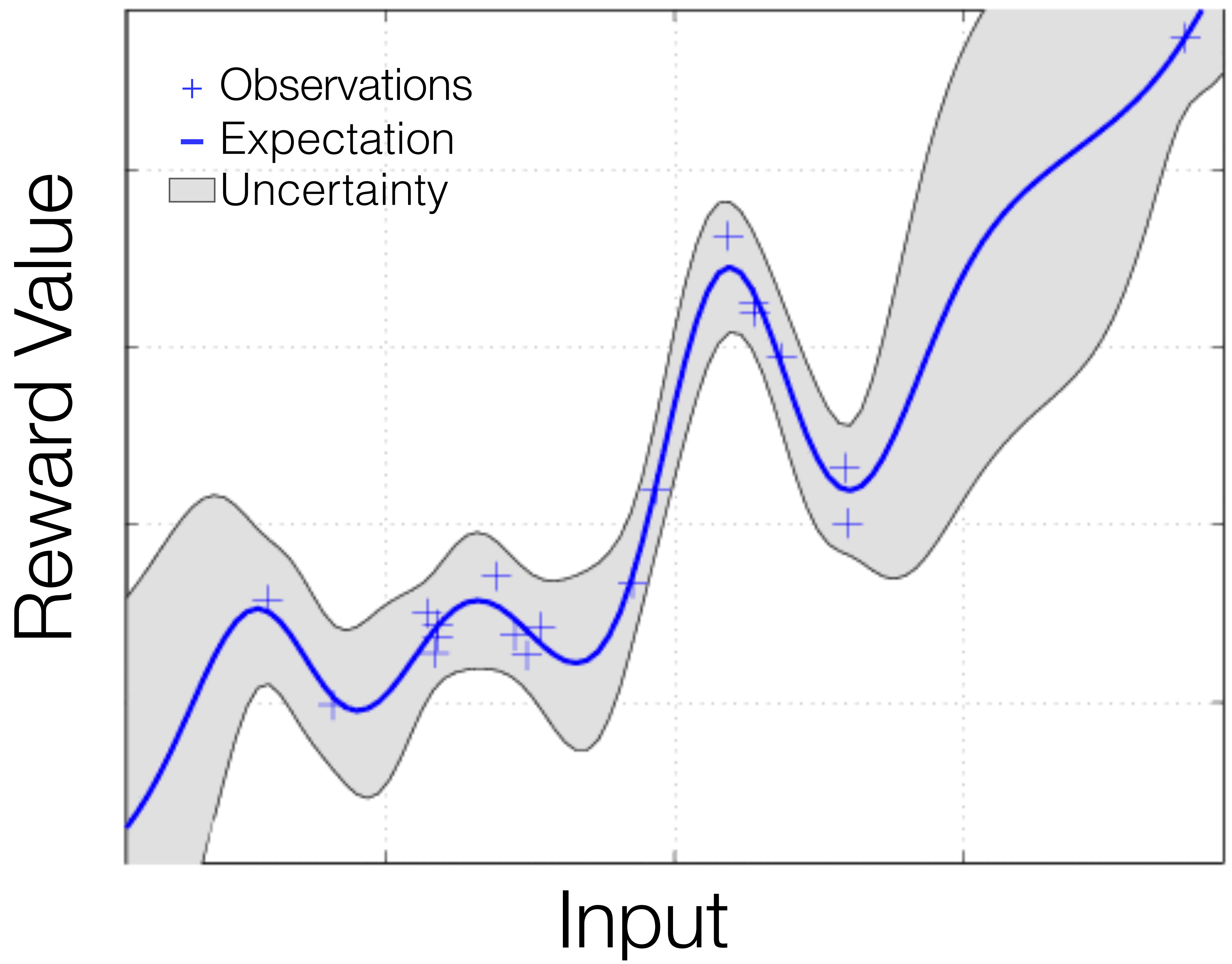




Bayesian Function Learning using Gaussian Process (GP) Regression



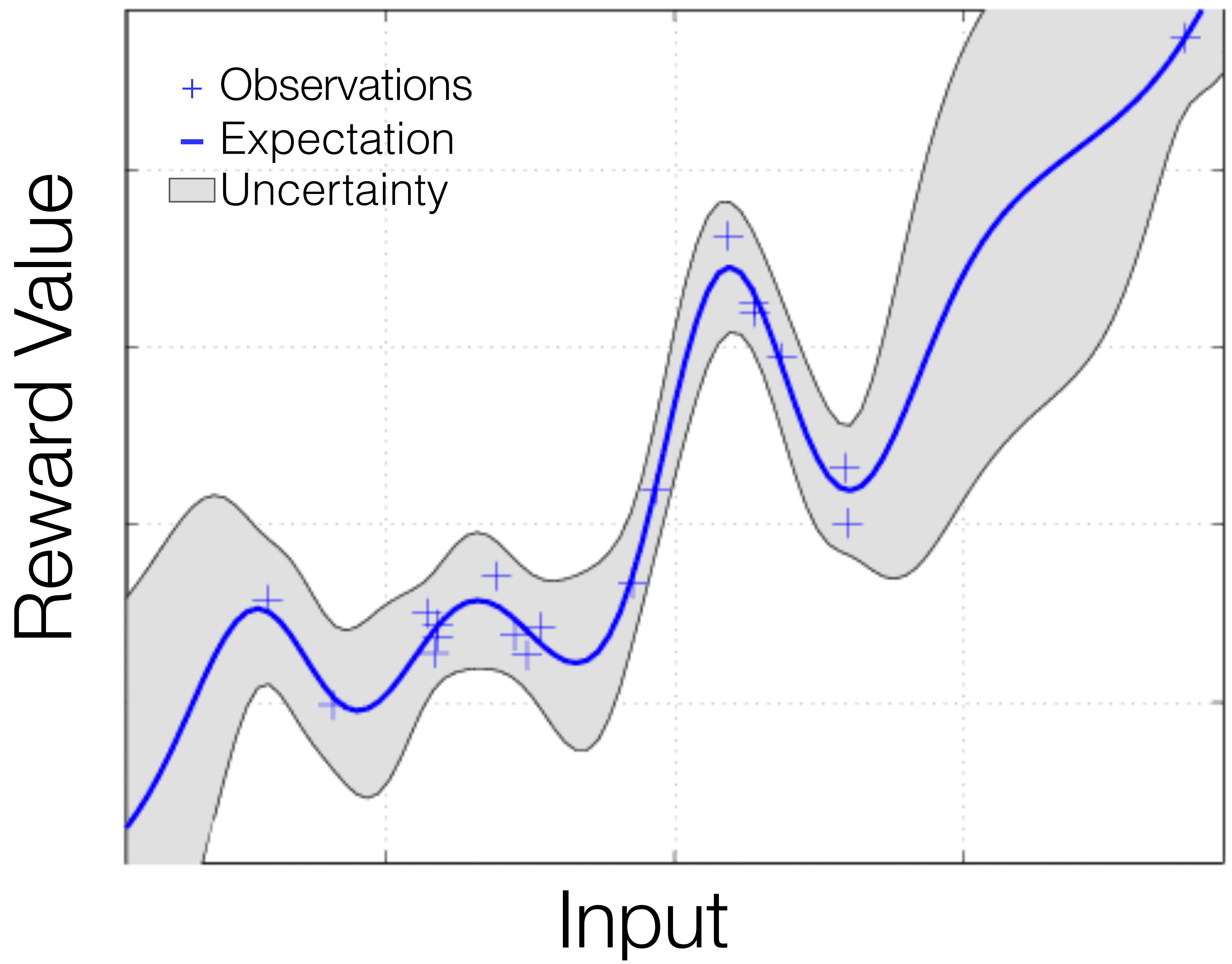
Bayesian Function Learning using Gaussian Process (GP) Regression



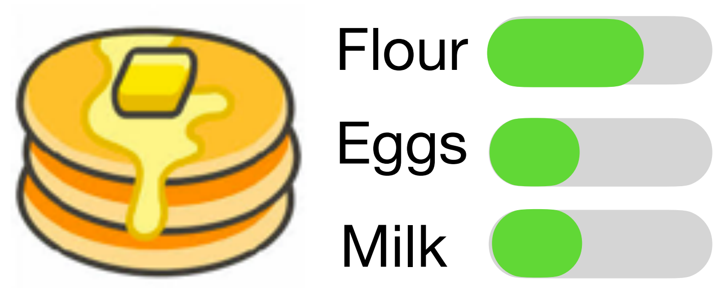
Location

(Wu et al., *NHB* 2018)

Bayesian Function Learning using Gaussian Process (GP) Regression



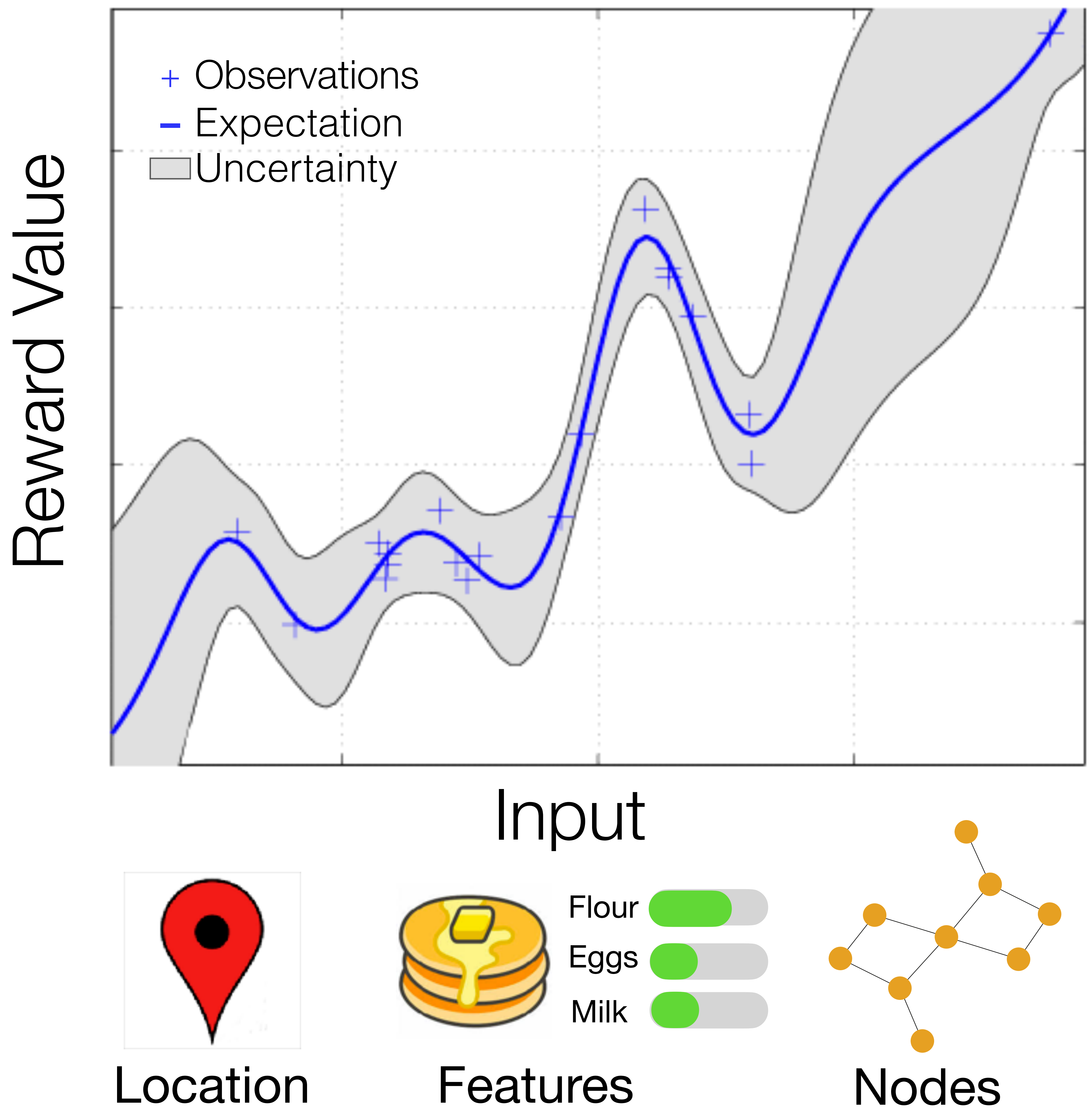
Location



Features

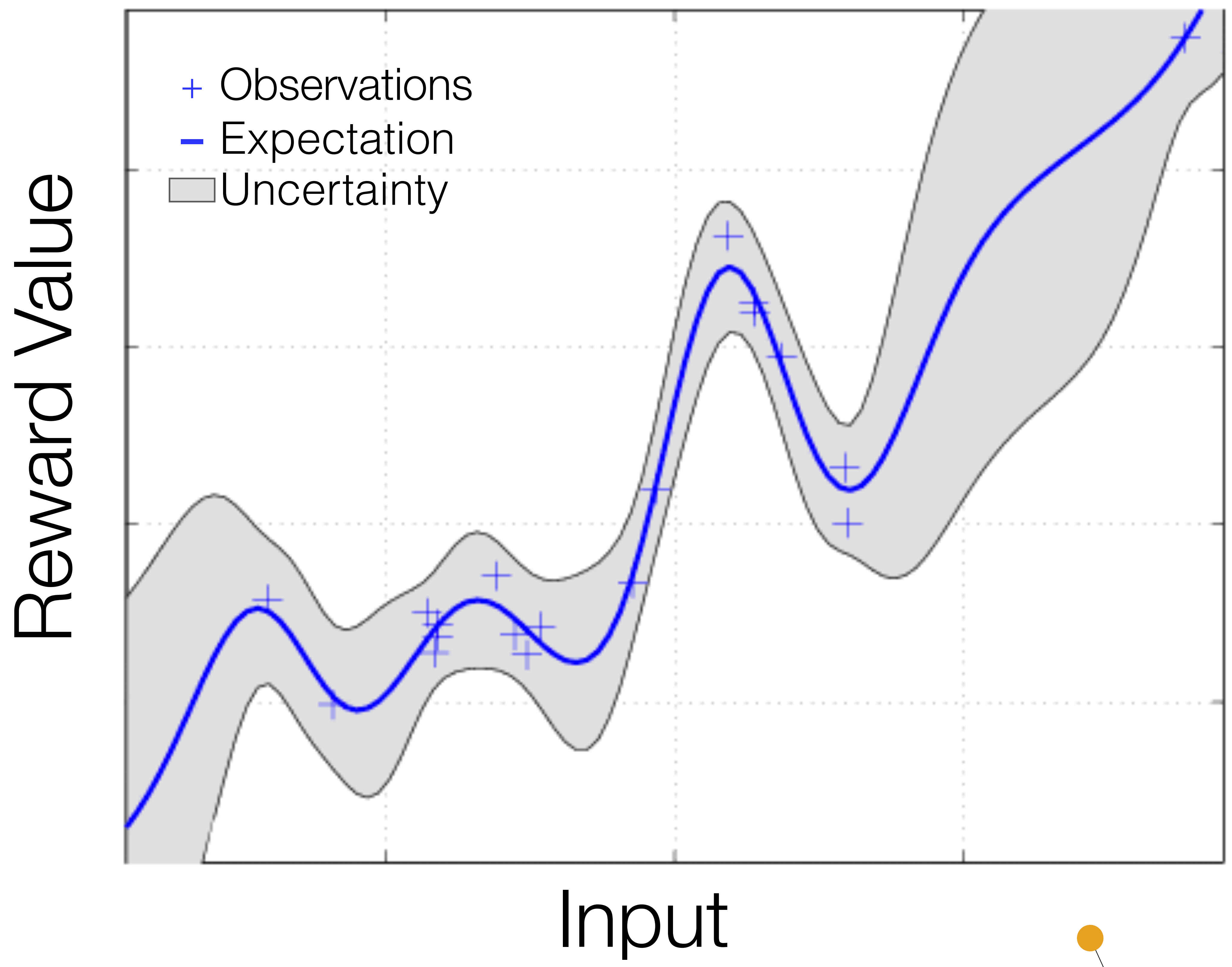
(Wu et al., *NHB* 2018)
(Wu et al., *PLOS CompBio* 2020)

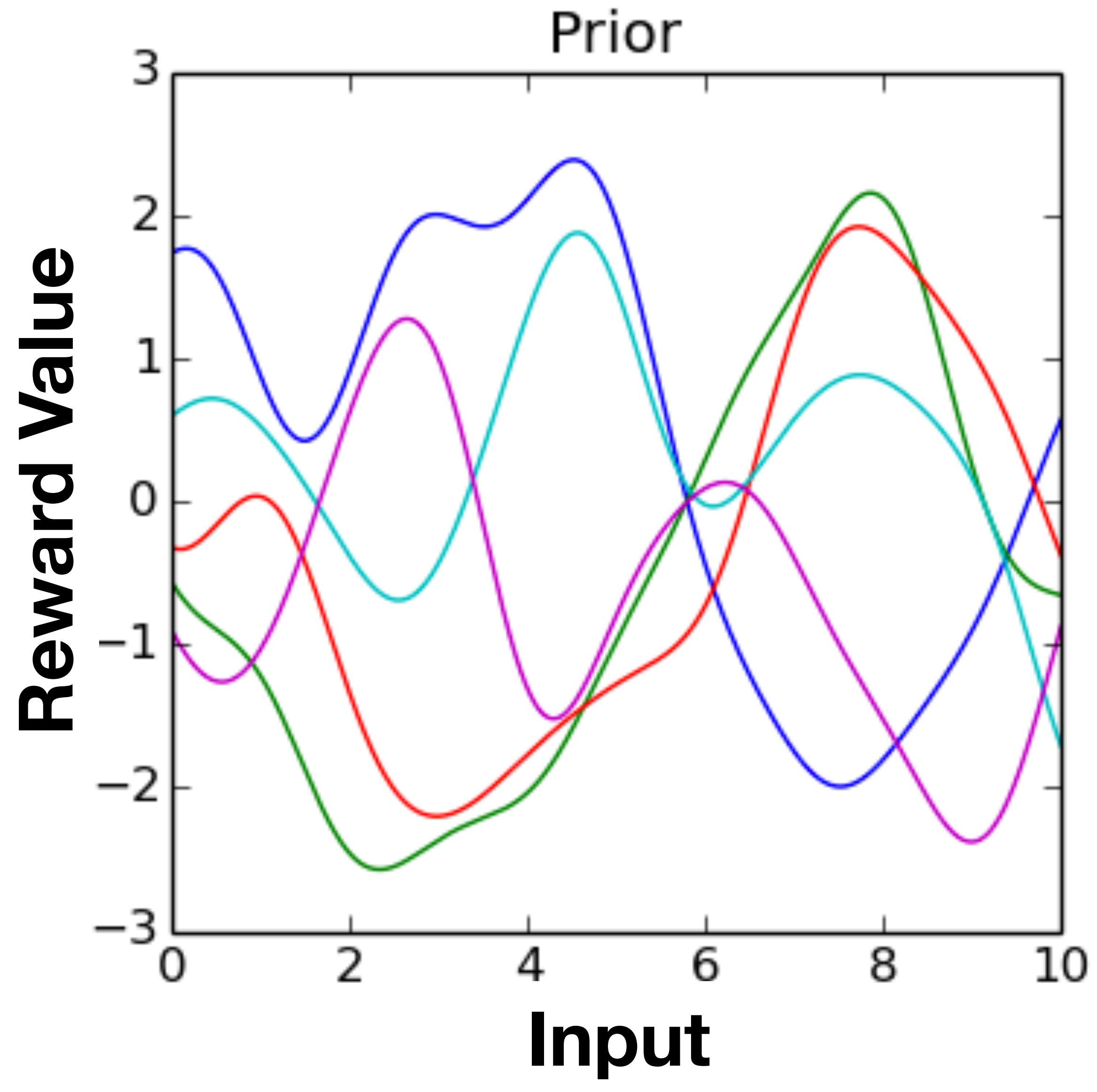
Bayesian Function Learning using Gaussian Process (GP) Regression

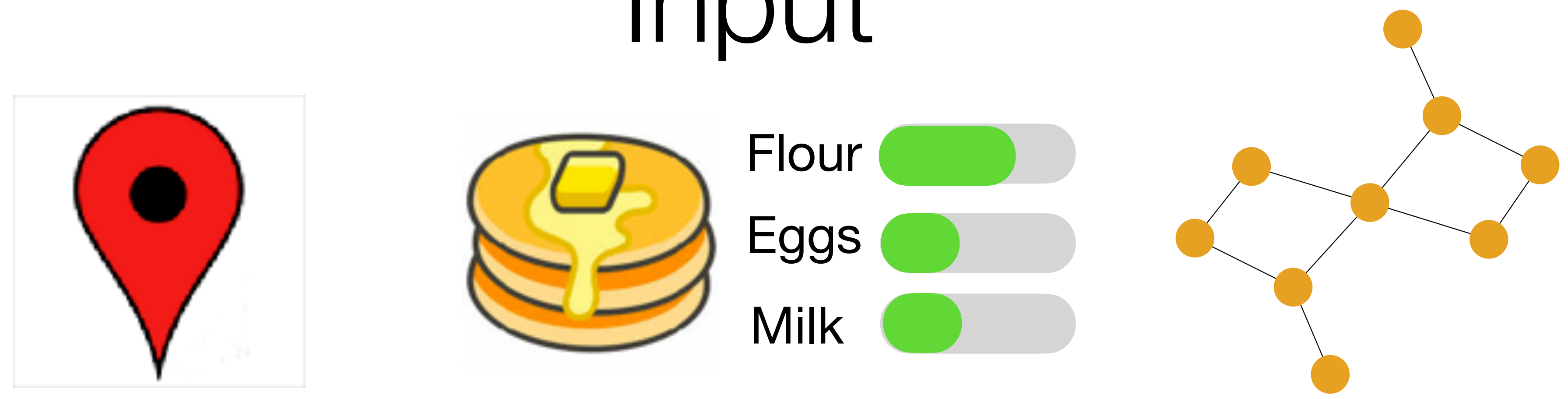


(Wu et al., *NHB* 2018)
(Wu et al., *PLOS CompBio* 2020)
(Wu et al., *CBB* 2021)

Bayesian Function Learning using Gaussian Process (GP) Regression



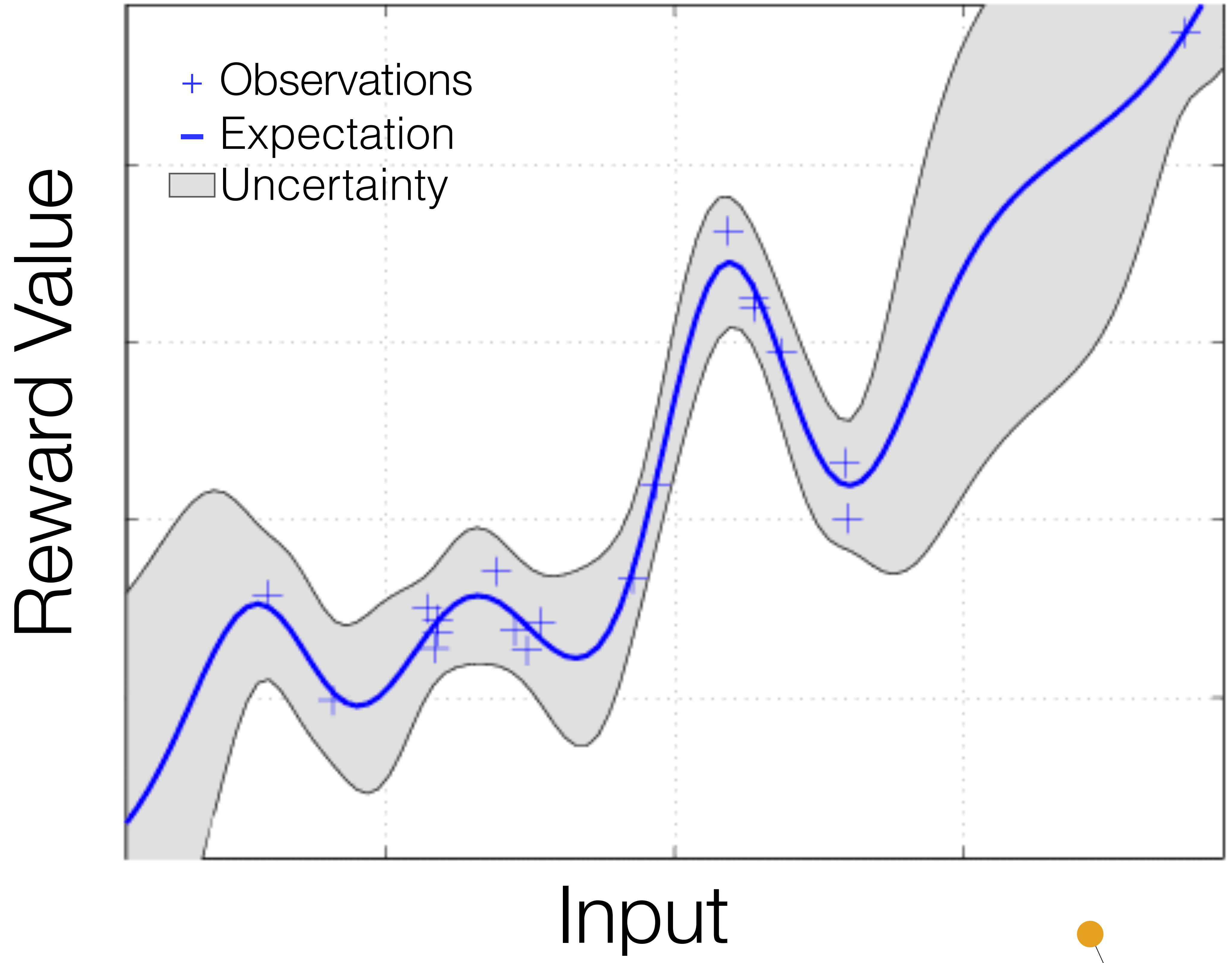


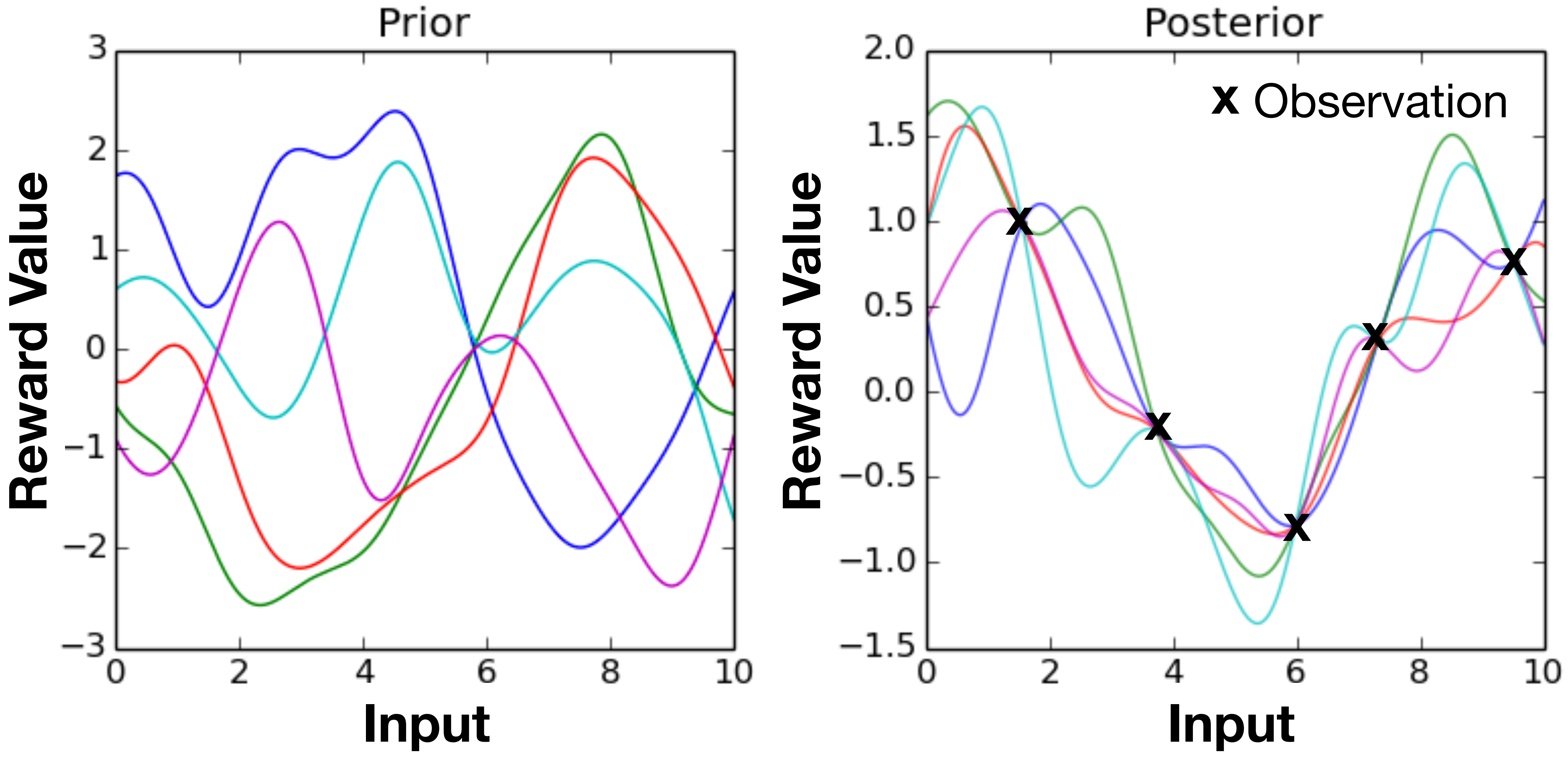


Location Features Nodes

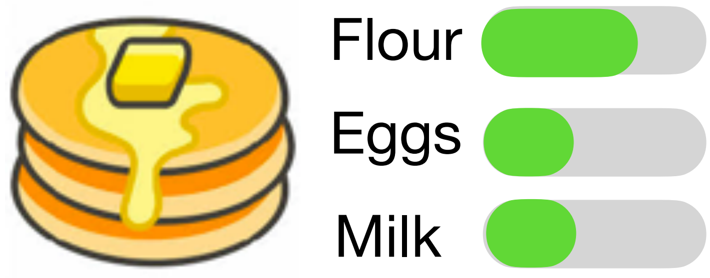
(Wu et al., *NHB* 2018)
(Wu et al., *PLOS CompBio* 2020)
(Wu et al., *CBB* 2021)

Bayesian Function Learning using Gaussian Process (GP) Regression

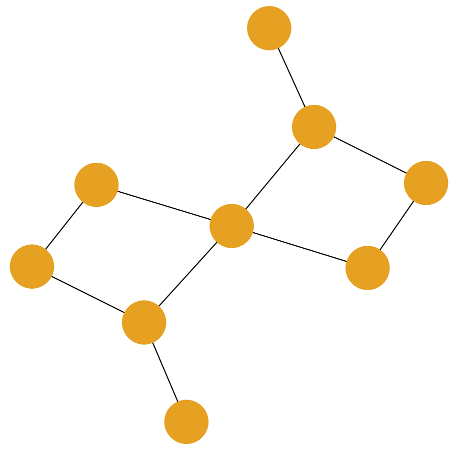




Location



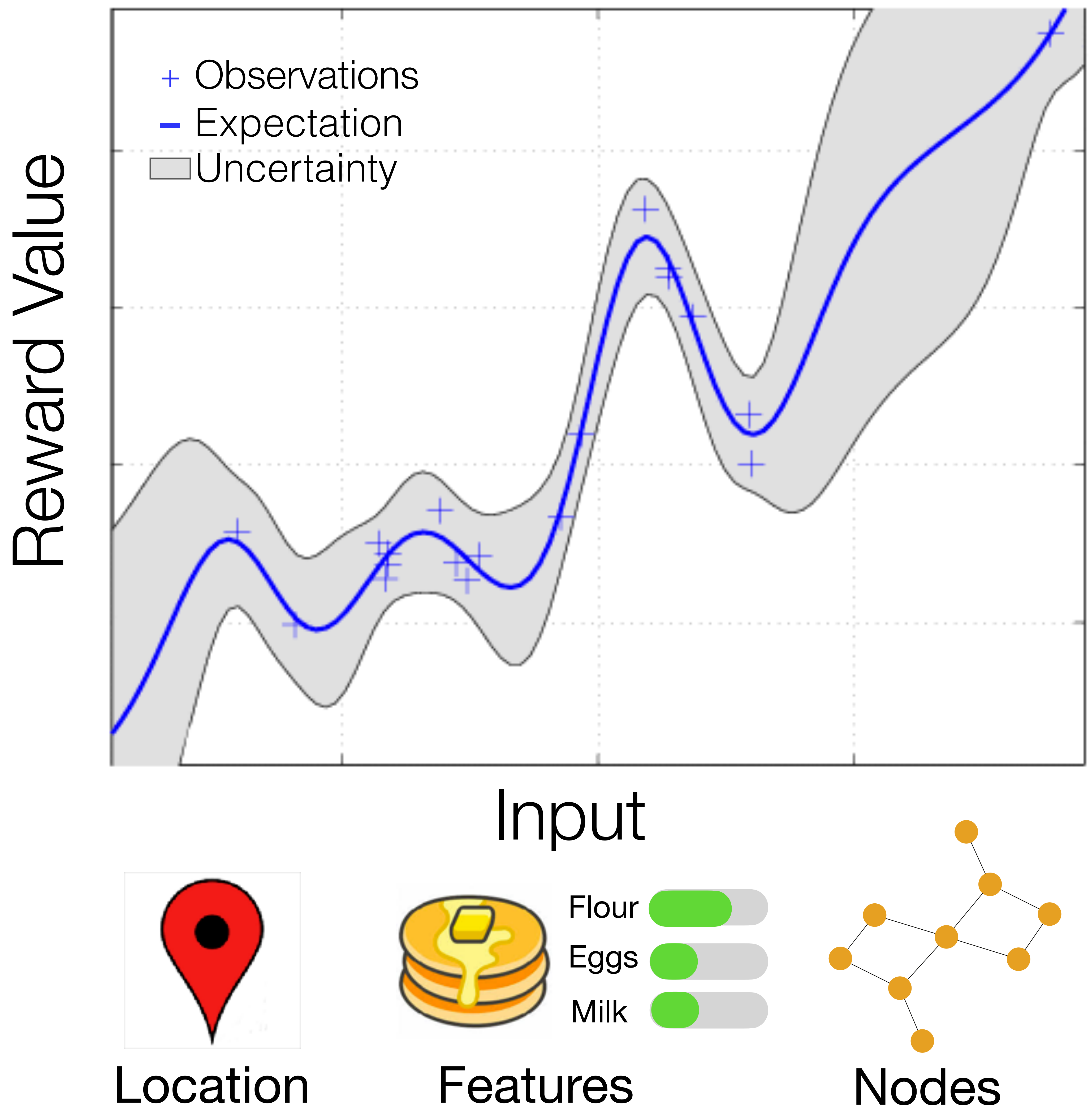
Features

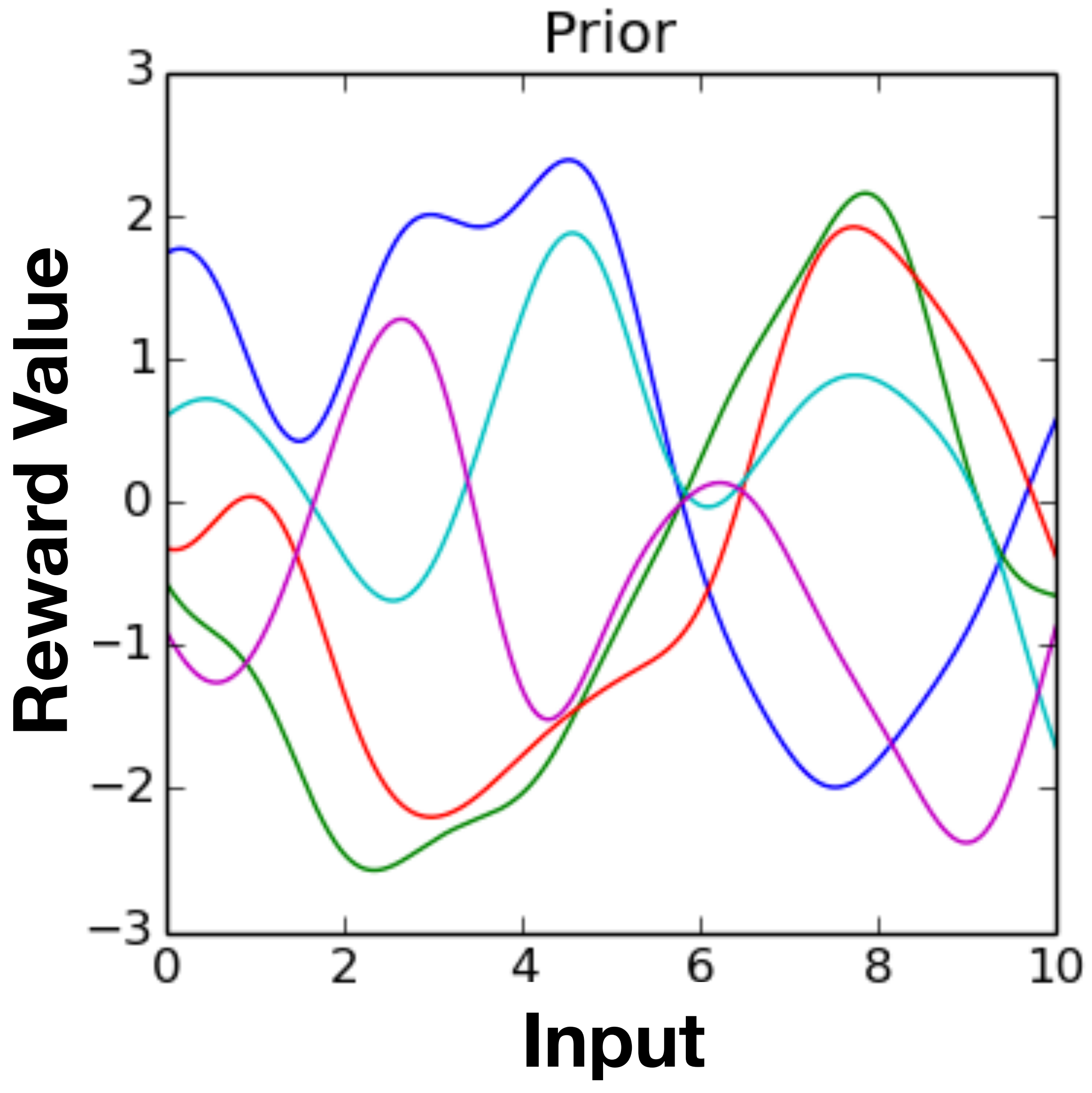


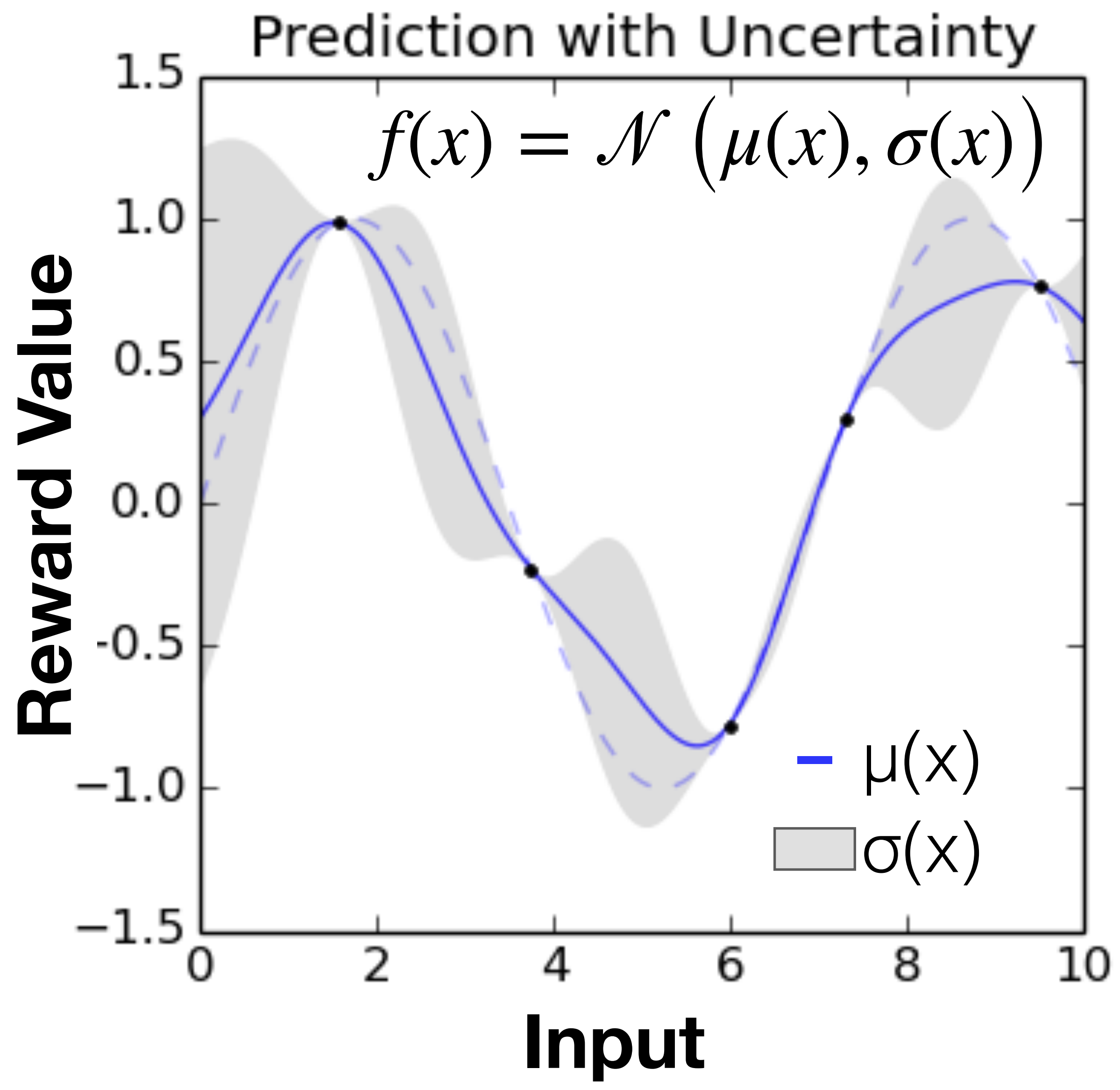
Nodes

(Wu et al., *NHB* 2018)
(Wu et al., *PLOS CompBio* 2020)
(Wu et al., *CBB* 2021)

Bayesian Function Learning using Gaussian Process (GP) Regression

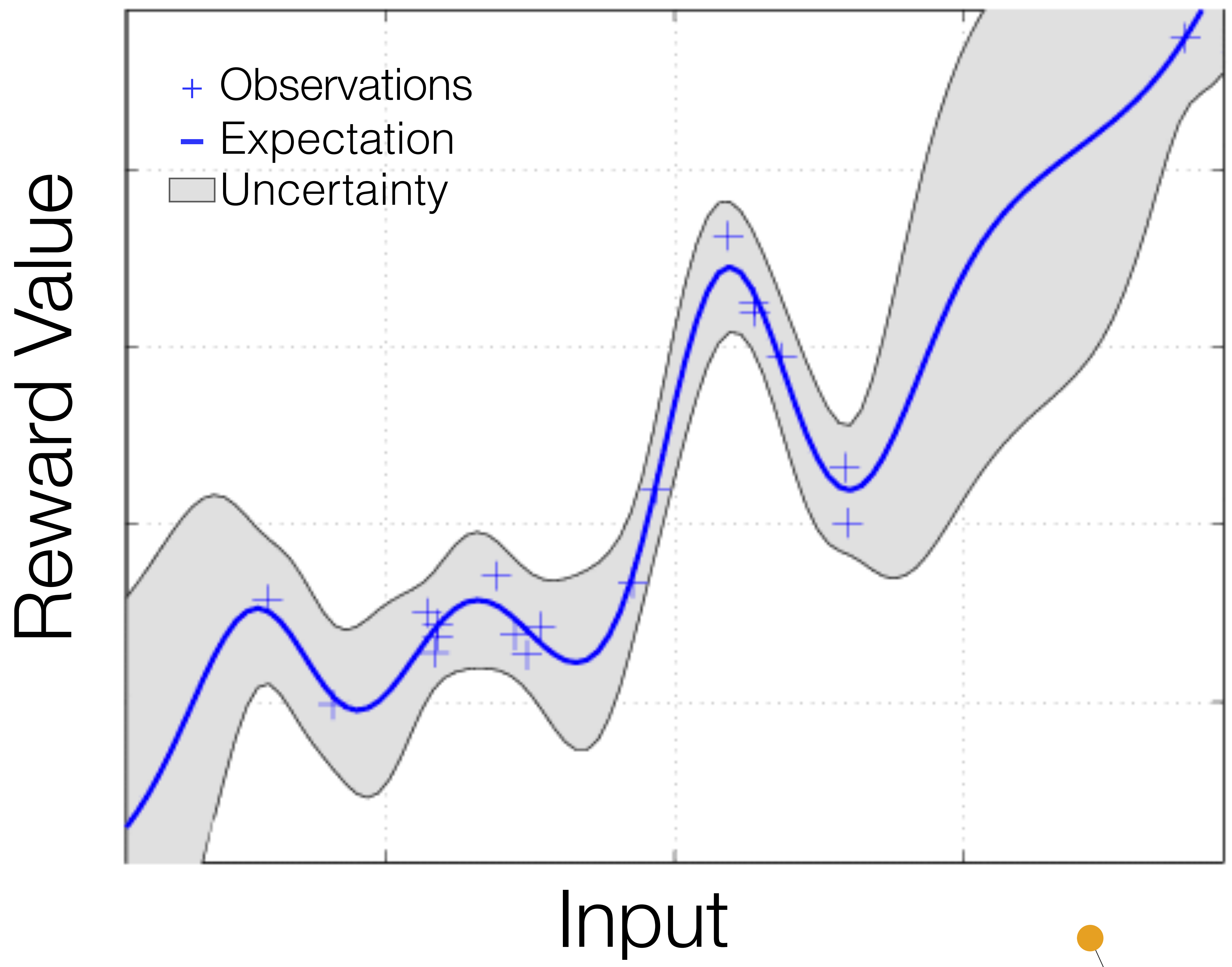


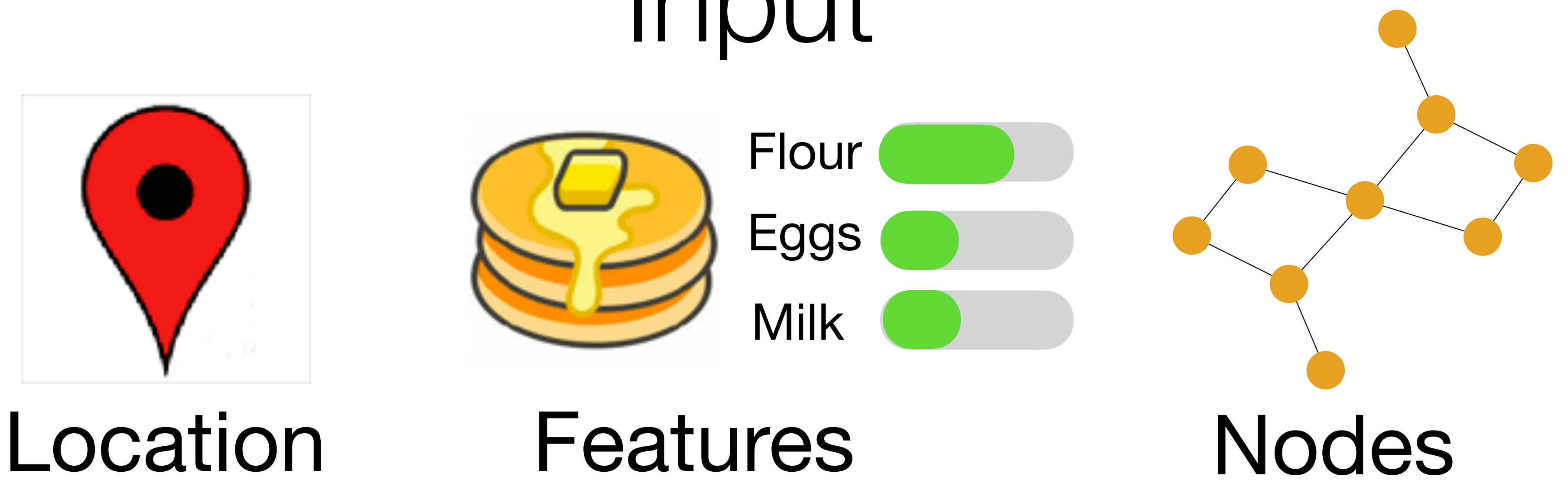




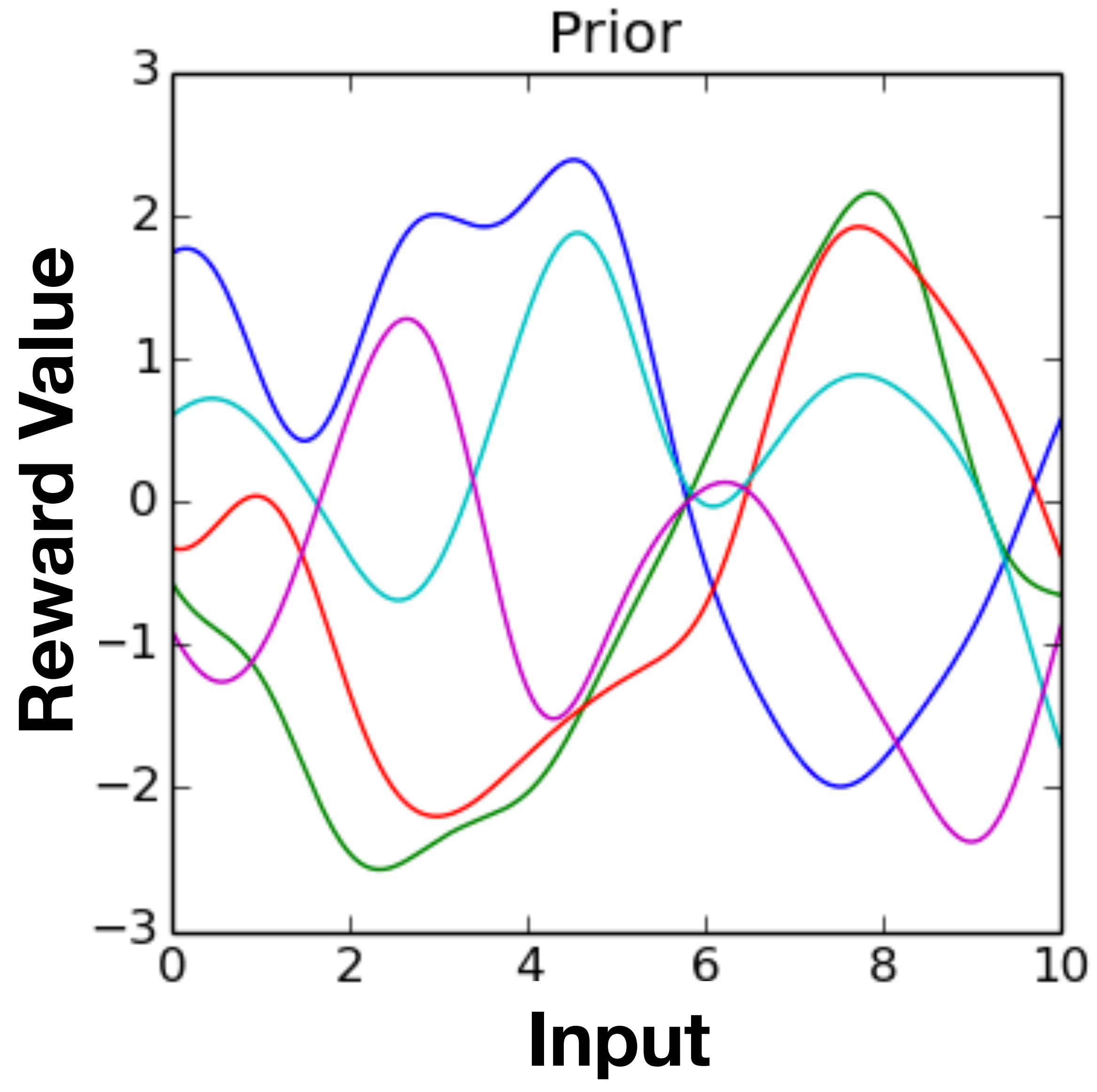
(Wu et al., *NHB* 2018)
(Wu et al., *PLOS CompBio* 2020)
(Wu et al., *CBB* 2021)

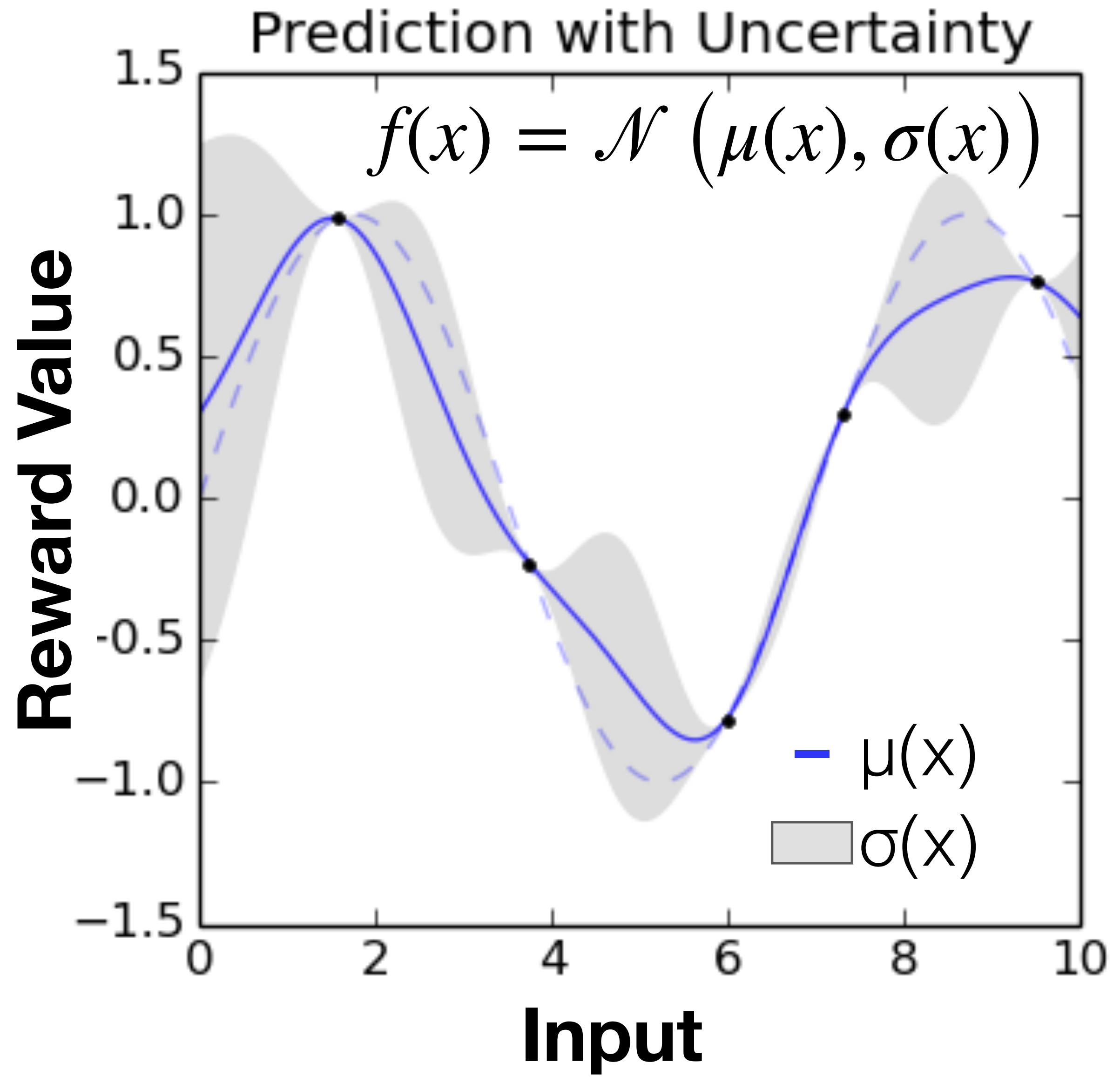
Bayesian Function Learning using Gaussian Process (GP) Regression

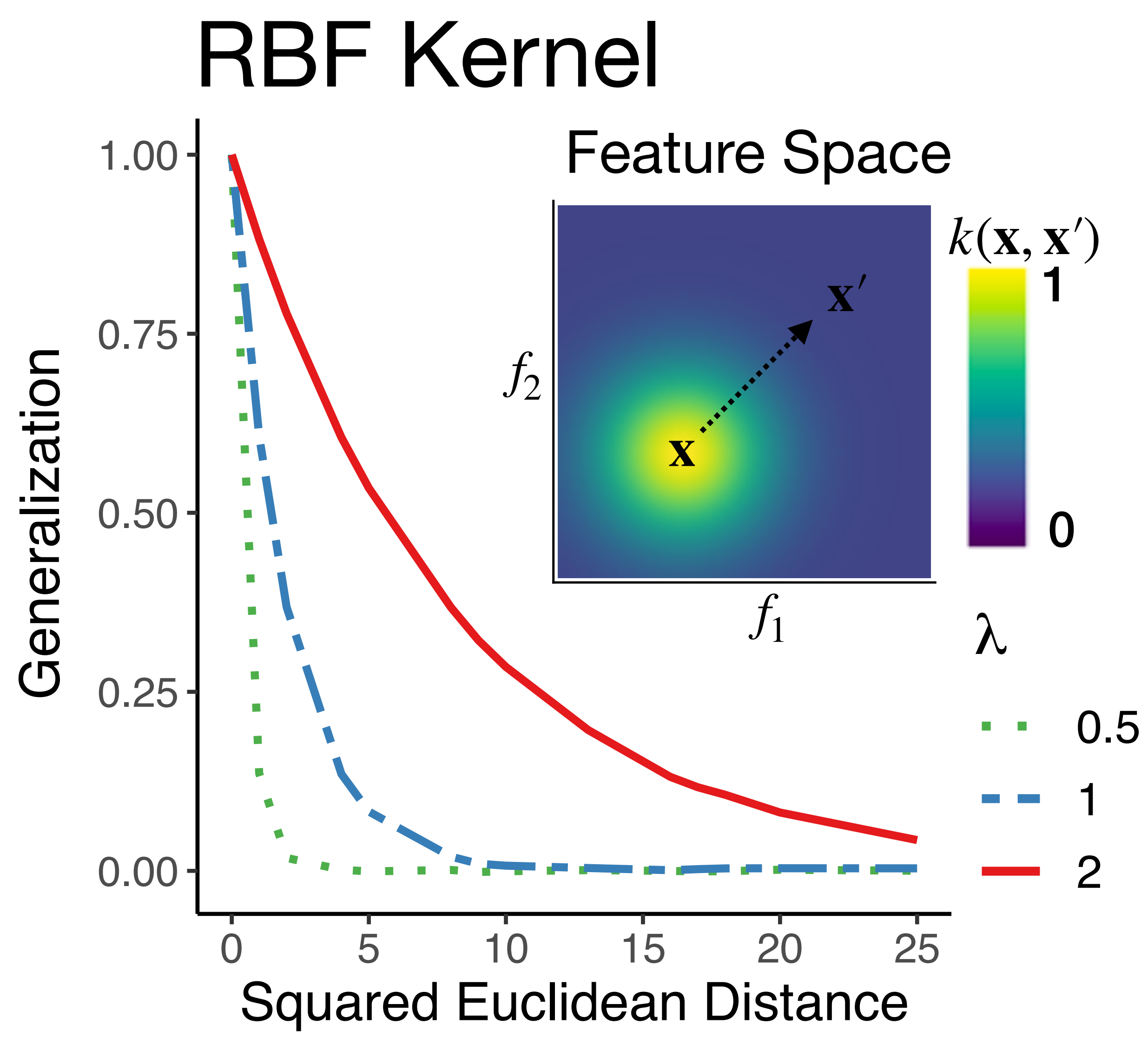


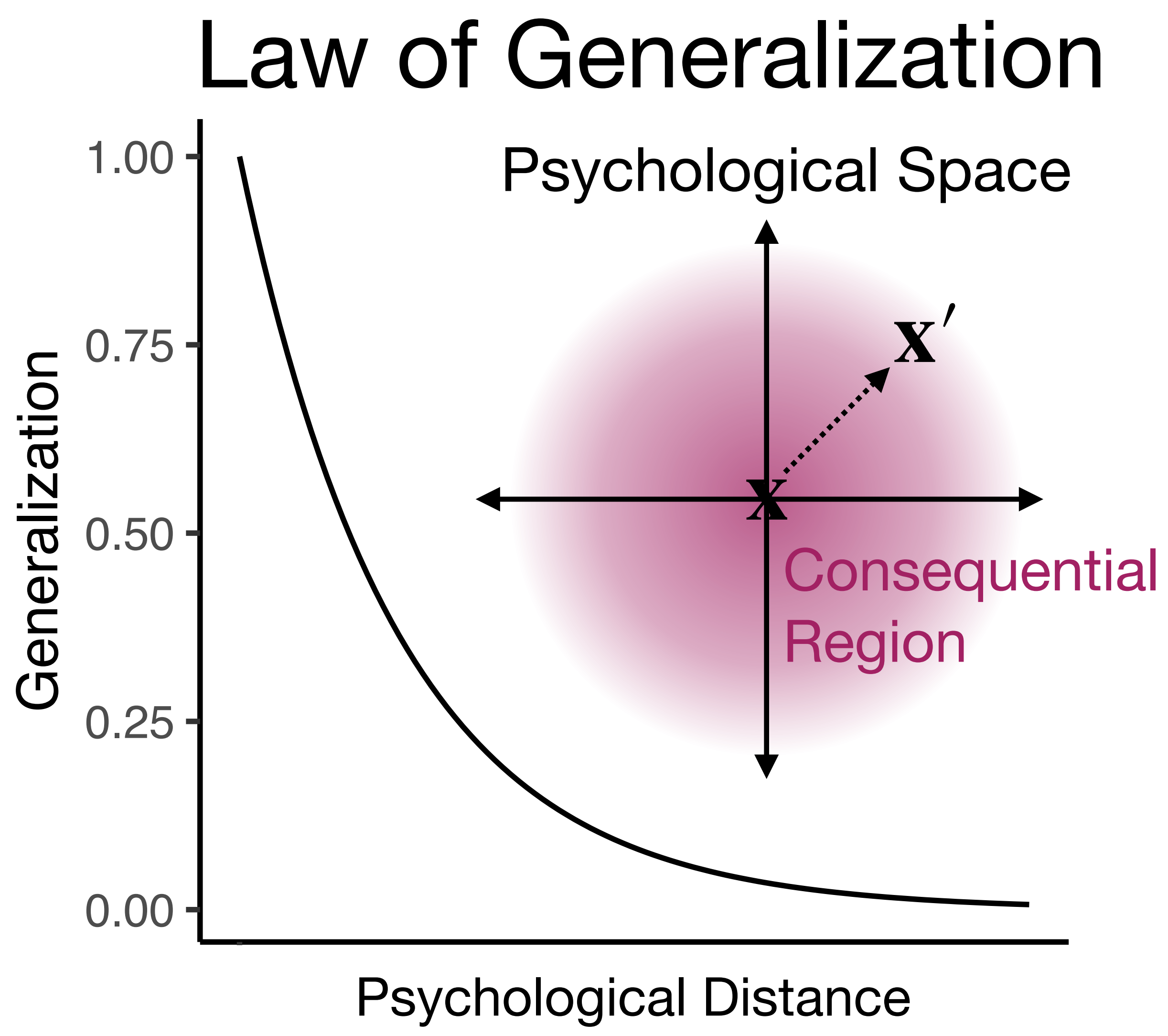


(Wu et al., *NHB* 2018)
(Wu et al., *PLOS CompBio* 2020)
(Wu et al., *CBB* 2021)



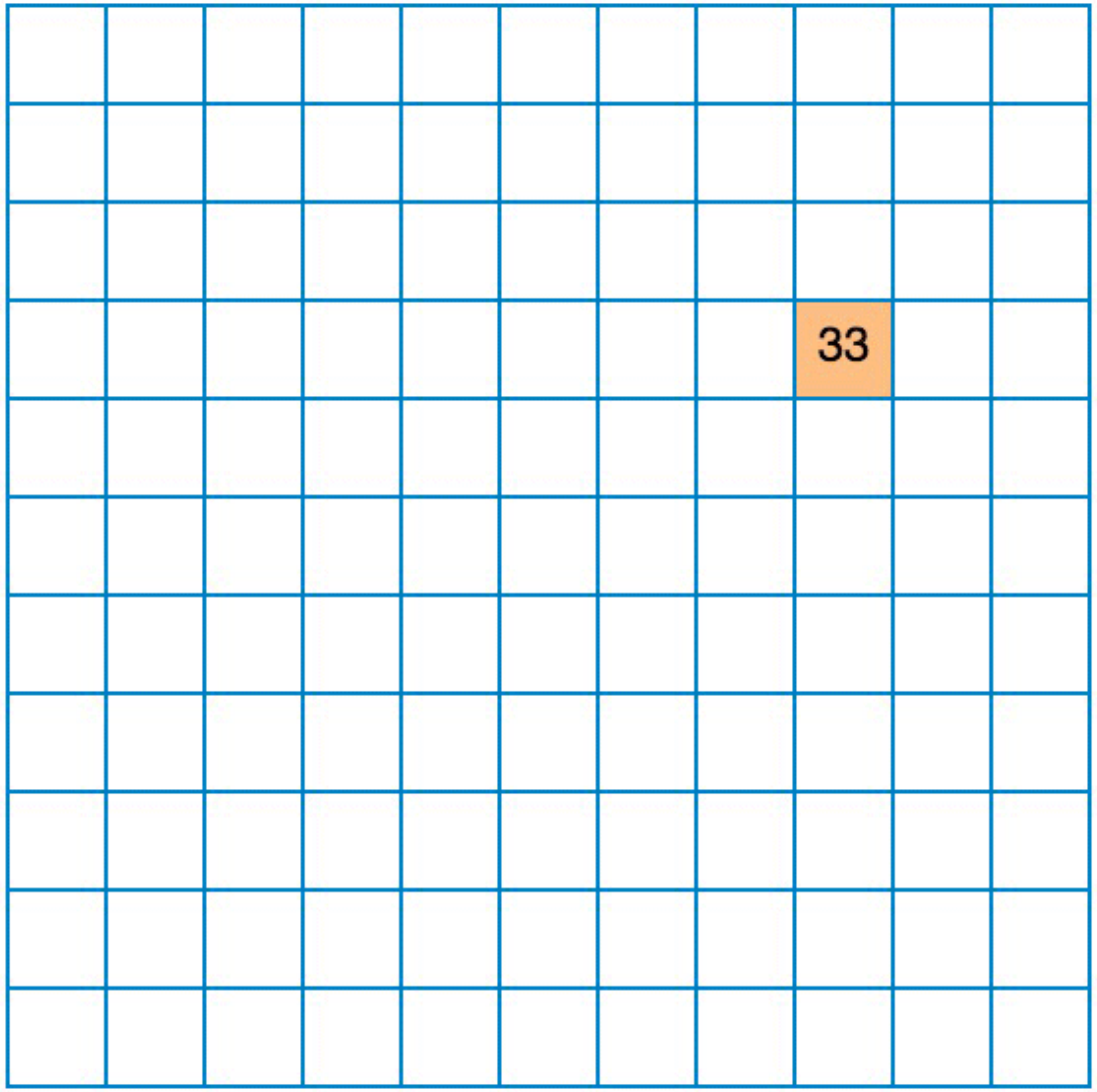






Wu et al., (*AnnRevPsy* in press)

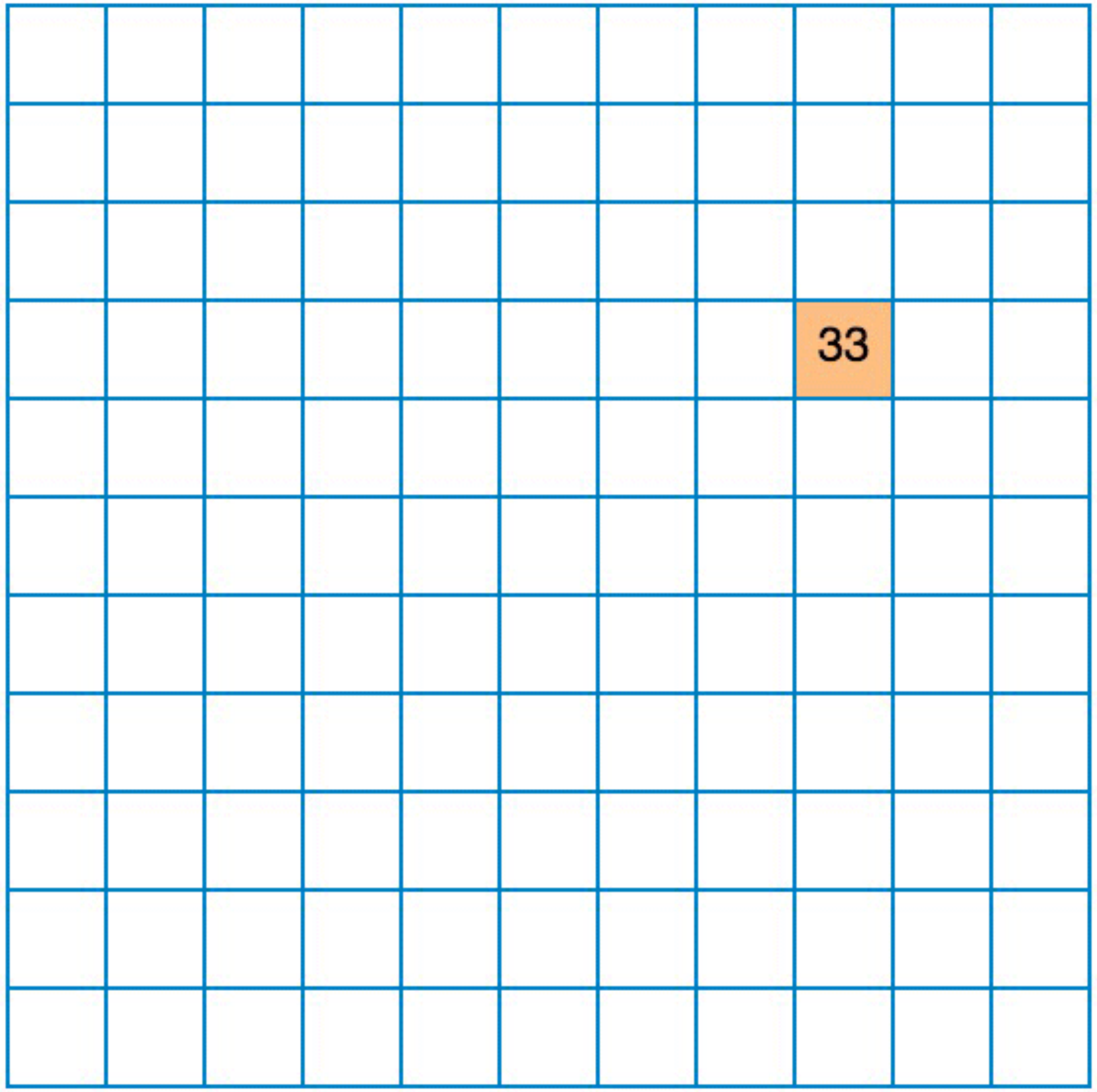
Spatially Correlated Bandit



Wu et al., (*Nature Human Behaviour* 2018)

- click tiles on the grid
- maximize reward
- each tile has normally distributed rewards
- limited search horizon
- nearby tiles have similar rewards

Spatially Correlated Bandit



Wu et al., (*Nature Human Behaviour* 2018)

- click tiles on the grid
- maximize reward
- each tile has normally distributed rewards
- limited search horizon
- nearby tiles have similar rewards

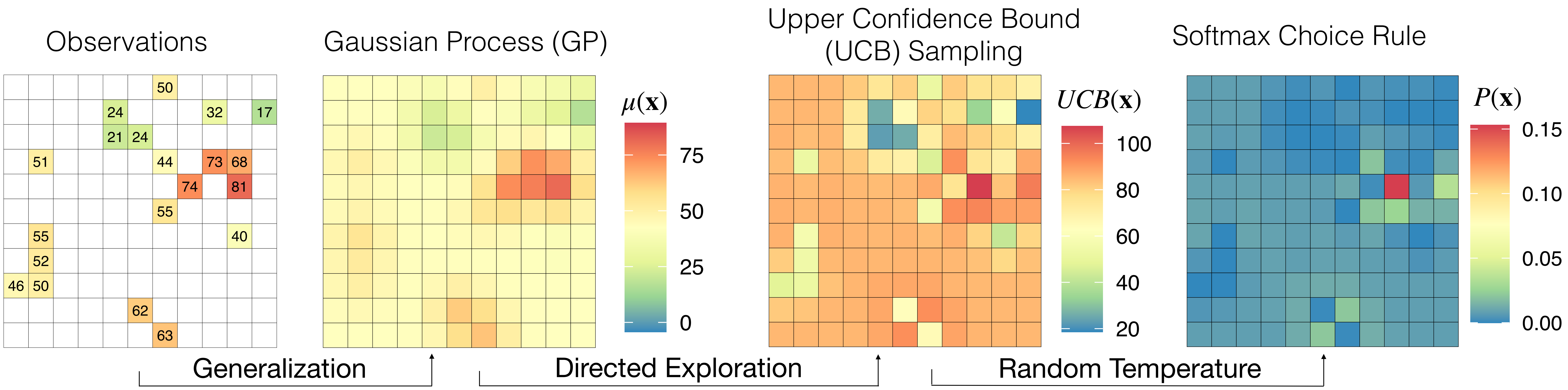
Spatially Correlated Bandit

7	5	10	22	32	32	28	24	22	26	33
6	11	19	29	38	41	42	40	37	36	40
22	27	30	35	43	50	53	53	51	49	46
45	44	38	36	40	46	47	49	54	55	48
61	55	46	40	37	32	27	31	44	52	44
62	59	57	54	44	27	14	17	33	46	45
53	59	68	71	59	36	17	15	28	45	51
46	57	71	77	67	47	26	18	27	45	56
45	56	65	67	60	46	29	20	27	42	55
51	57	58	53	47	40	30	23	28	40	49
60	62	58	47	39	38	35	31	35	41	46

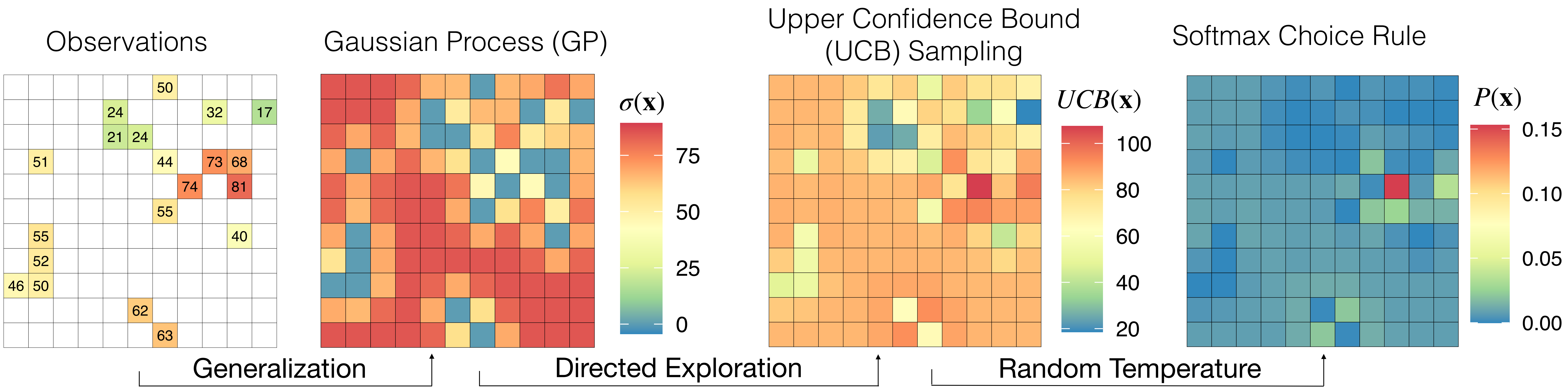
Wu et al., (*Nature Human Behaviour* 2018)

- click tiles on the grid
- maximize reward
- each tile has normally distributed rewards
- limited search horizon
- nearby tiles have similar rewards

GP-UCB Model

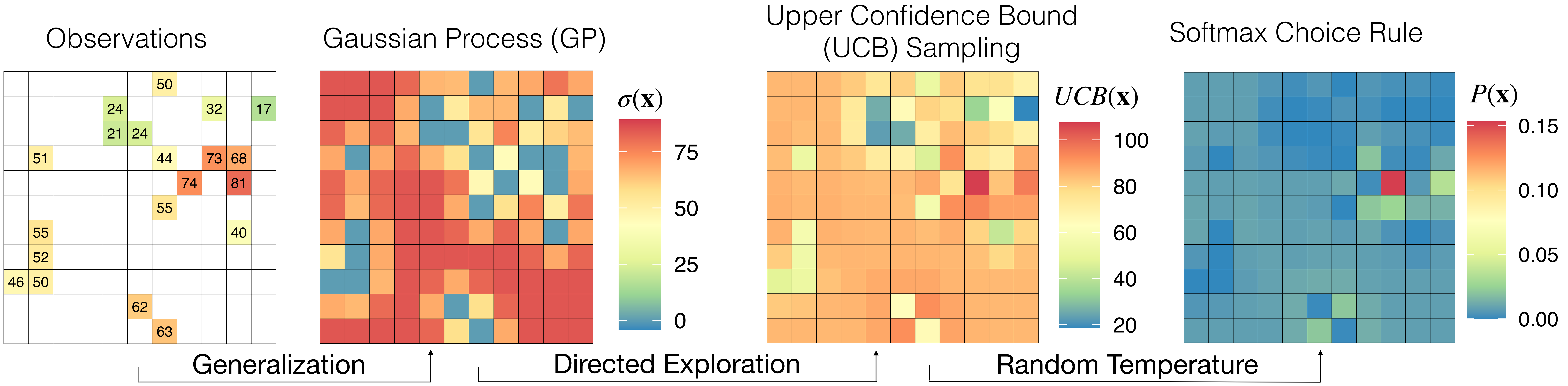

Observations
Gaussian Process (GP)
Upper Confidence Bound (UCB) Sampling
Softmax Choice Rule
50
24
32
17
21
24
51
44
73
68
74
81
55
55
40
52
46
50
62
63
$\mu(\mathbf{x})$
75
50
25
0
$UCB(\mathbf{x})$
100
80
60
40
20
$P(\mathbf{x})$
0.15
0.10
0.05
0.00
Generalization
Directed Exploration
Random Temperature

GP-UCB Model

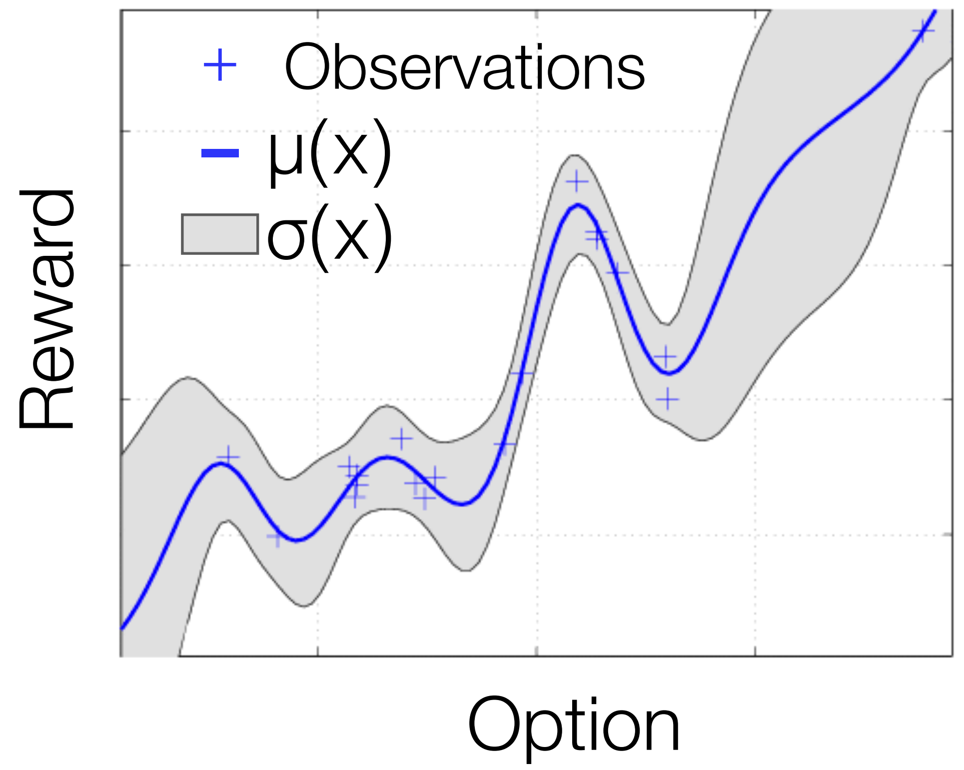


Observations
Gaussian Process (GP)
Upper Confidence Bound (UCB) Sampling
Softmax Choice Rule
50
24
32
17
21
24
51
44
73
68
74
81
55
55
40
52
46
50
62
63
σ(x)
75
50
25
0
UCB(x)
100
80
60
40
20
P(x)
0.15
0.10
0.05
0.00
Generalization
Directed Exploration
Random Temperature

GP-UCB Model

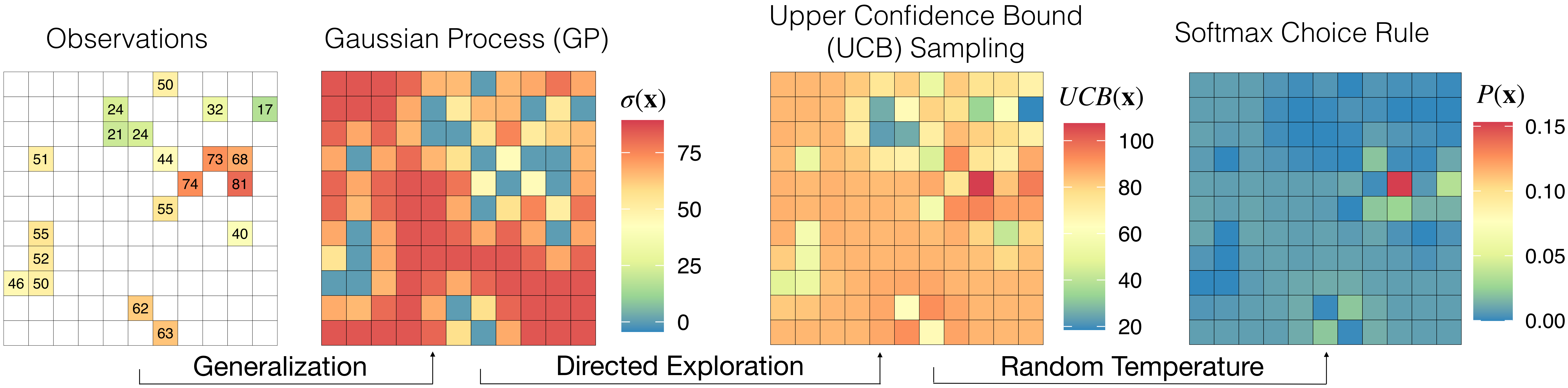


Observations
Gaussian Process (GP)
$\sigma(\mathbf{x})$
75
50
25
0
Upper Confidence Bound (UCB) Sampling
$UCB(\mathbf{x})$
100
80
60
40
20
Softmax Choice Rule
$P(\mathbf{x})$
0.15
0.10
0.05
0.00
Generalization
Directed Exploration
Random Temperature

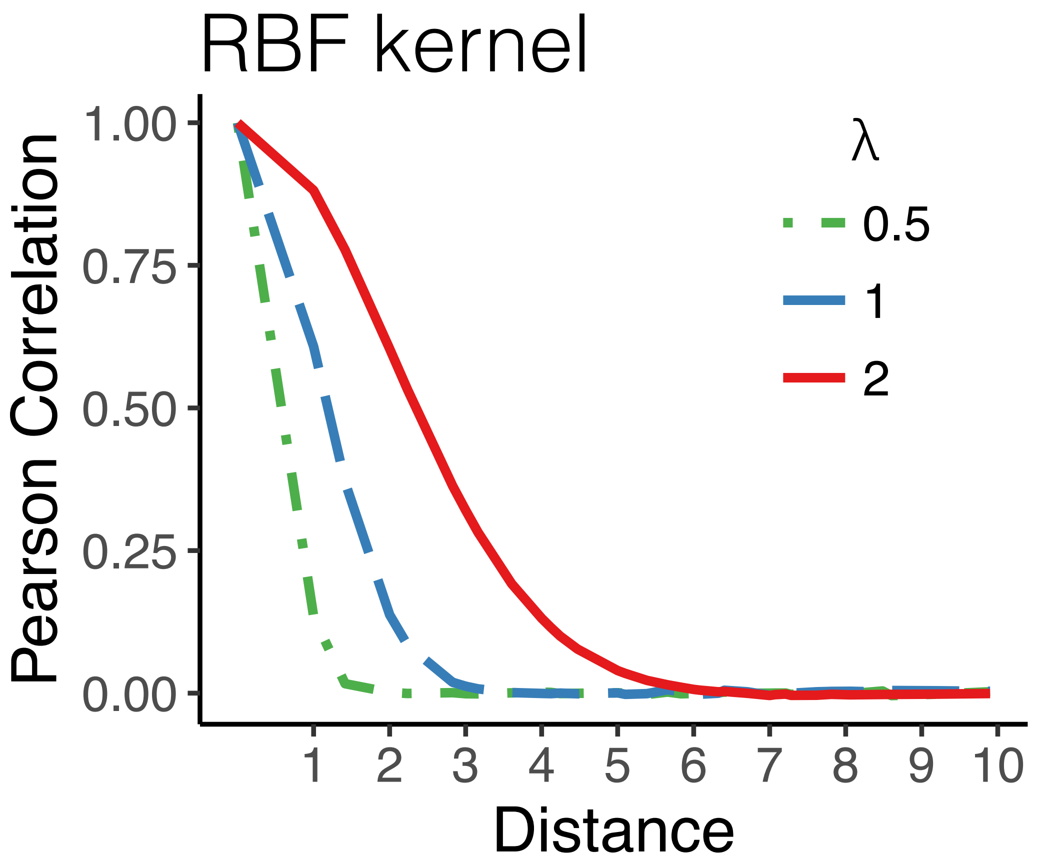


Observations
μ(x)
σ(x)
Reward
Option

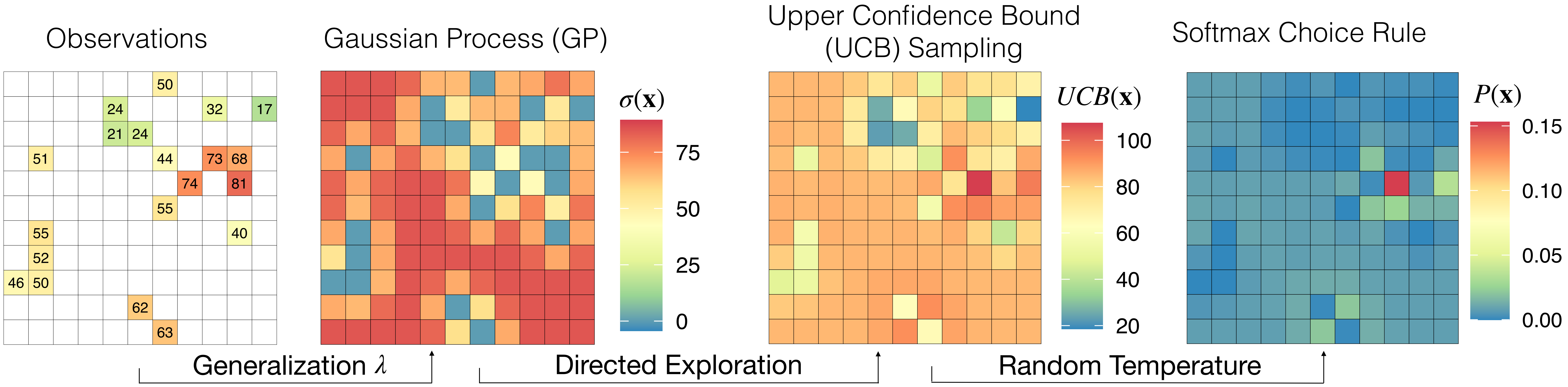
GP-UCB Model



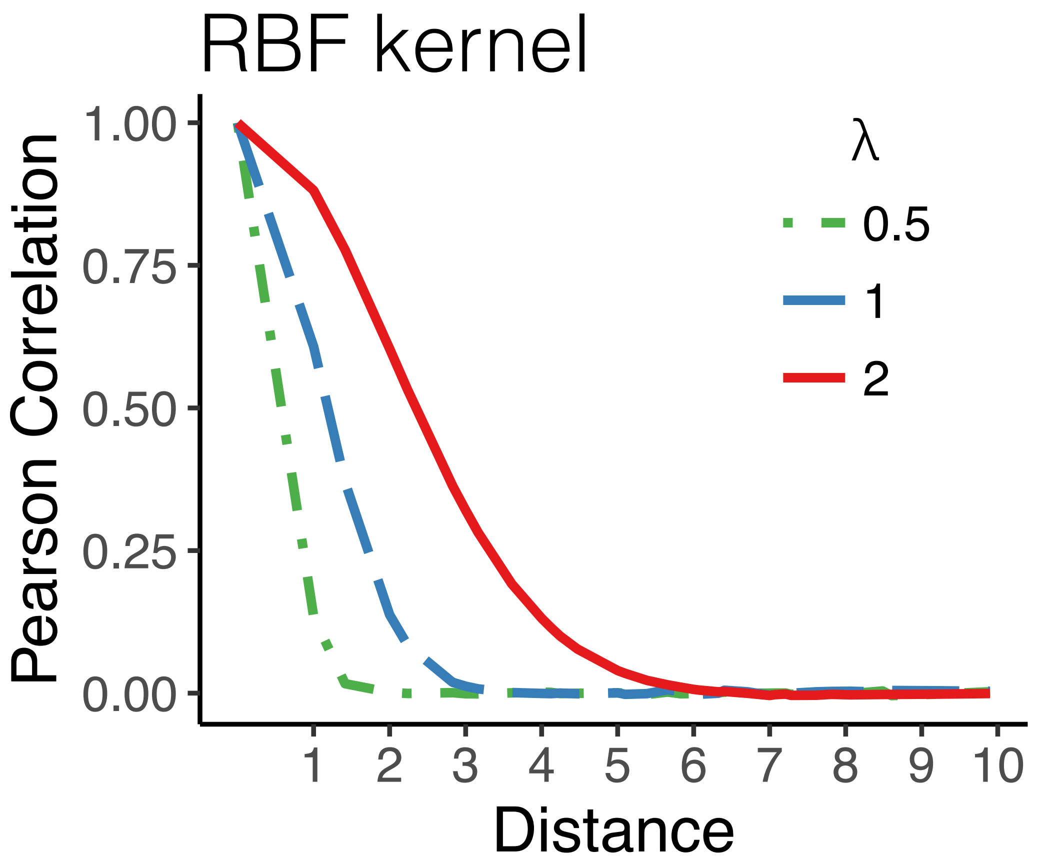
$$k_{RBF}(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{||\mathbf{x} - \mathbf{x}'||^2}{2\lambda^2}\right)$$



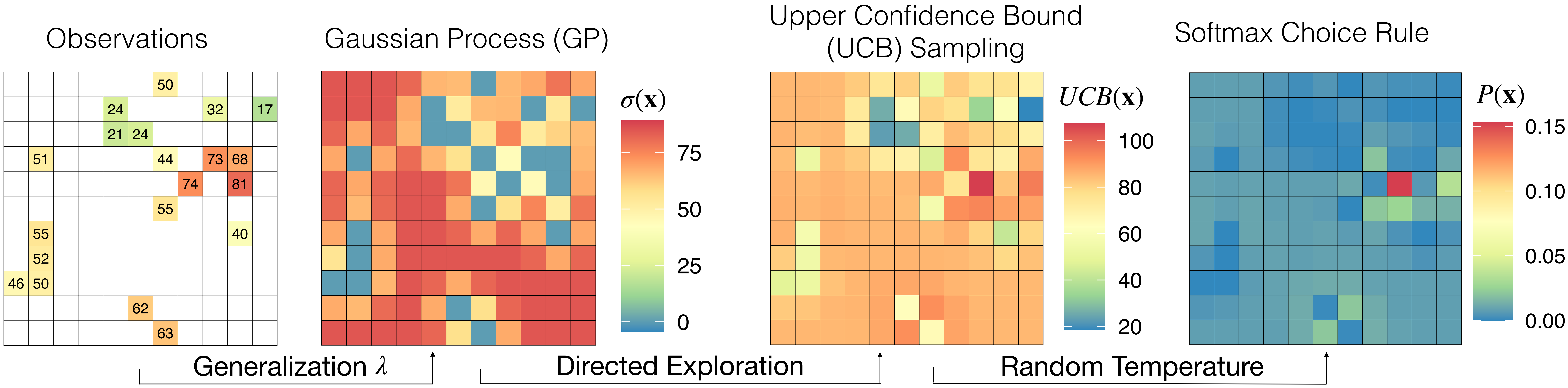
GP-UCB Model



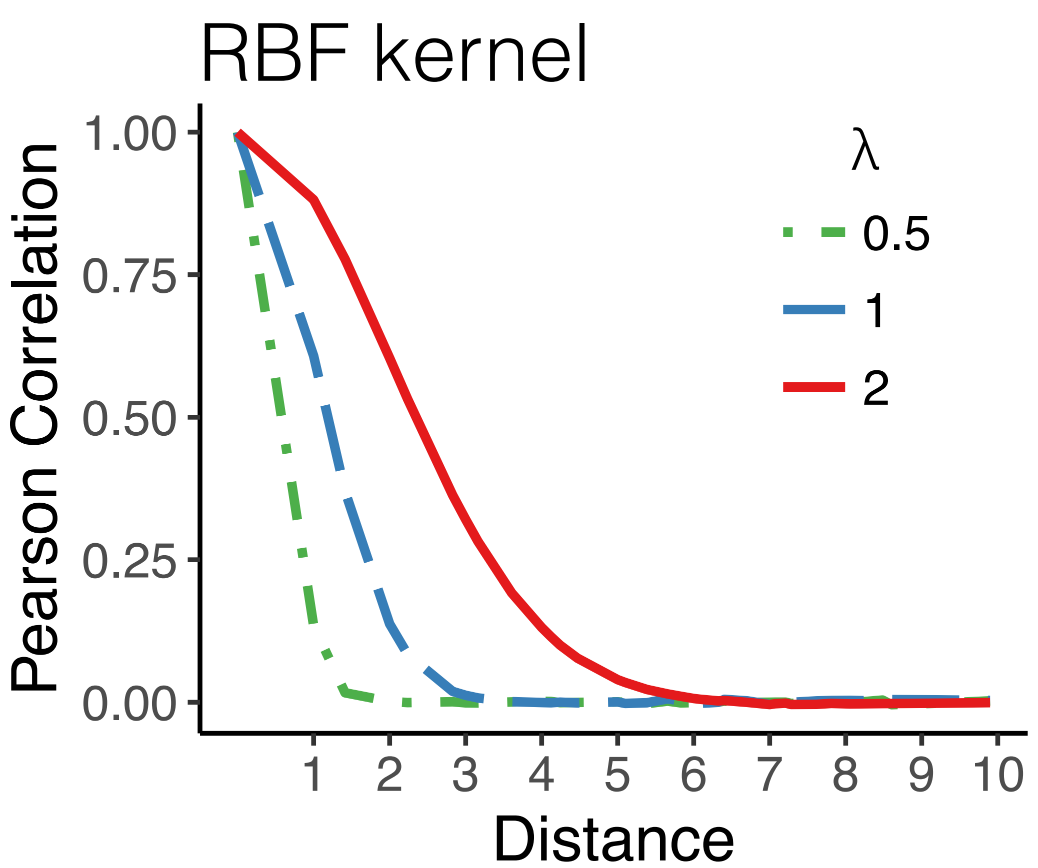
$$k_{RBF}(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{||\mathbf{x} - \mathbf{x}'||^2}{2\lambda^2}\right)$$

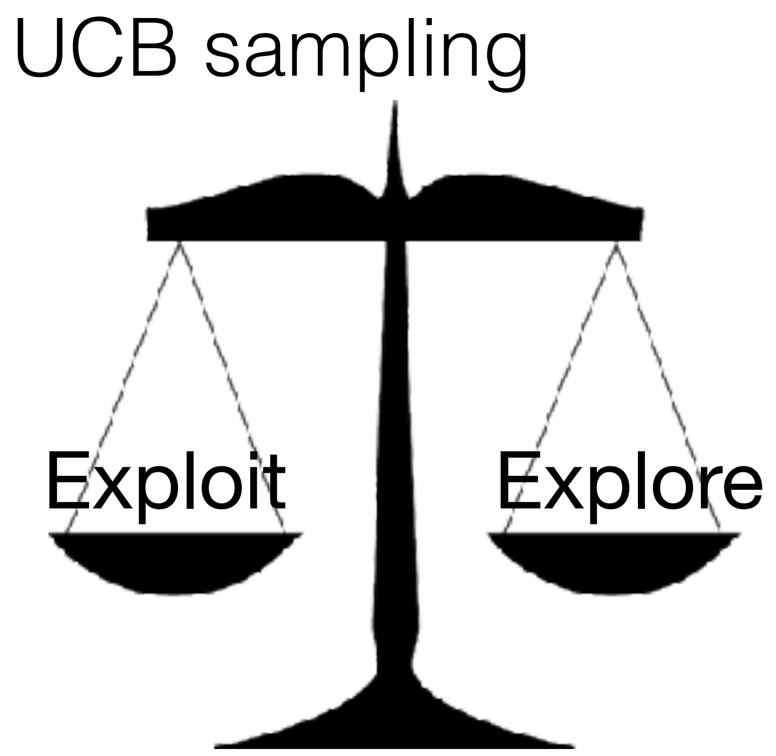


GP-UCB Model

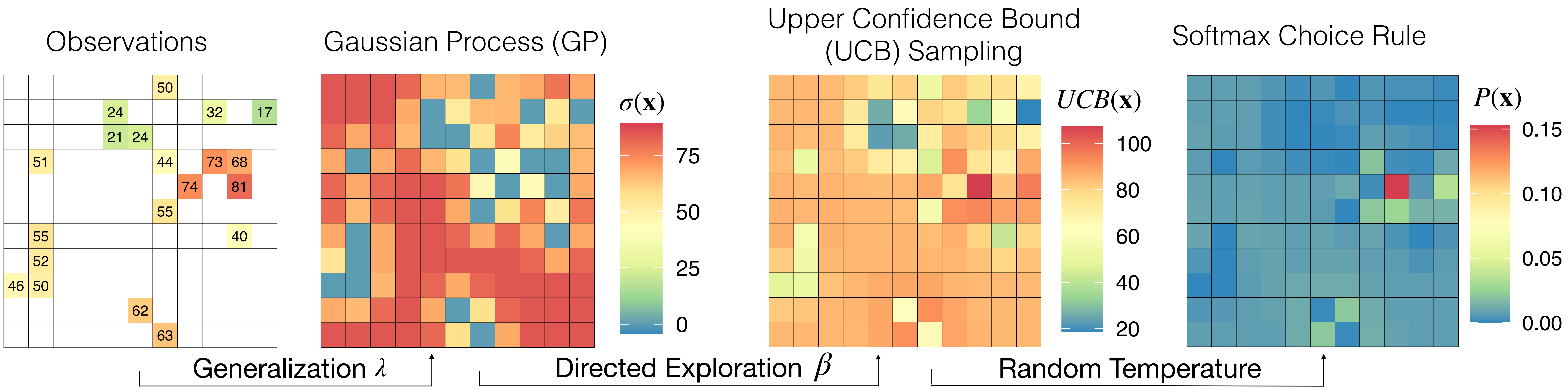


$$k_{RBF}(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{||\mathbf{x} - \mathbf{x}'||^2}{2\lambda^2}\right) \qquad UCB(\mathbf{x}) = \mu(\mathbf{x}) + \beta\sigma(\mathbf{x})$$

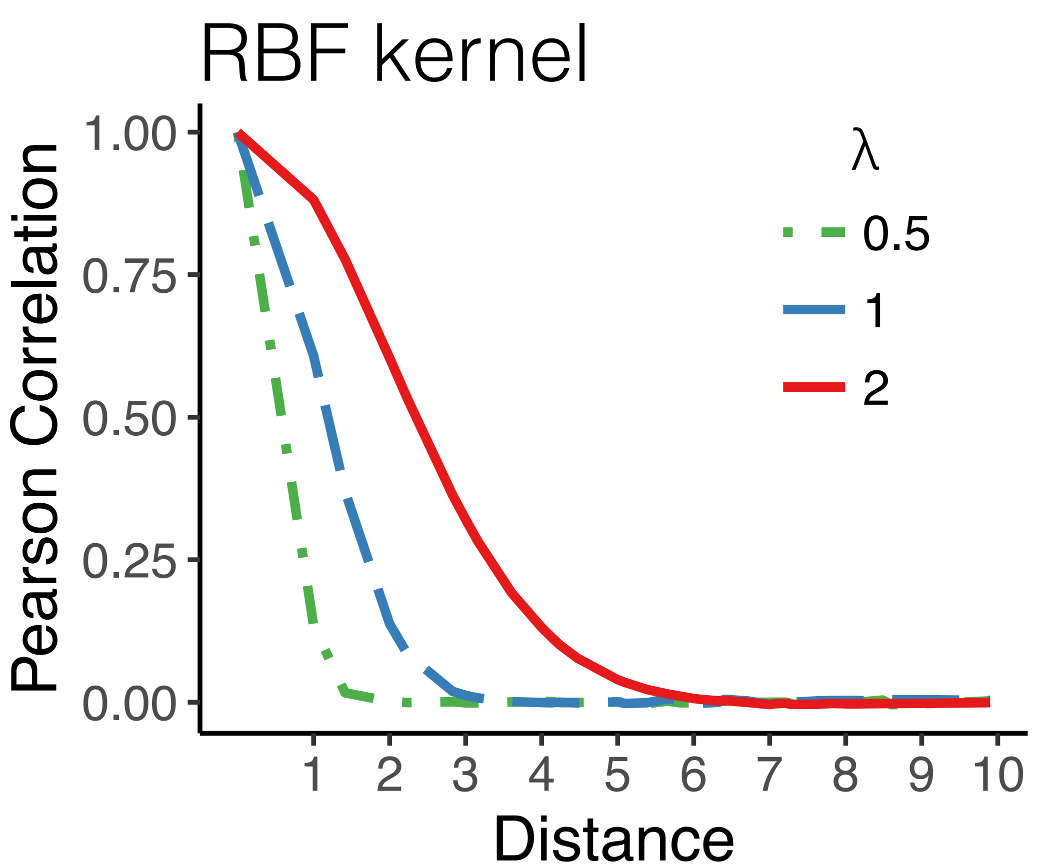


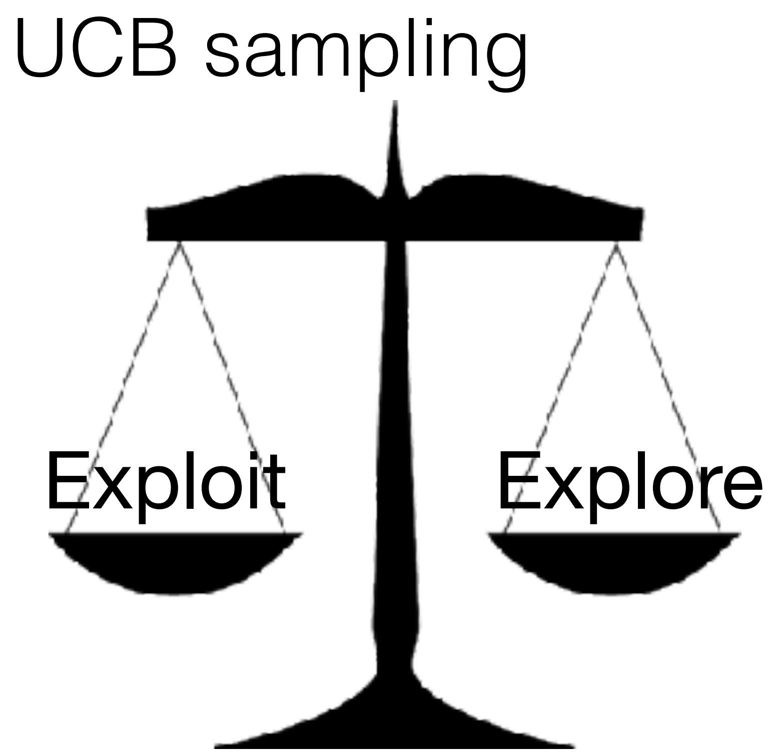


GP-UCB Model

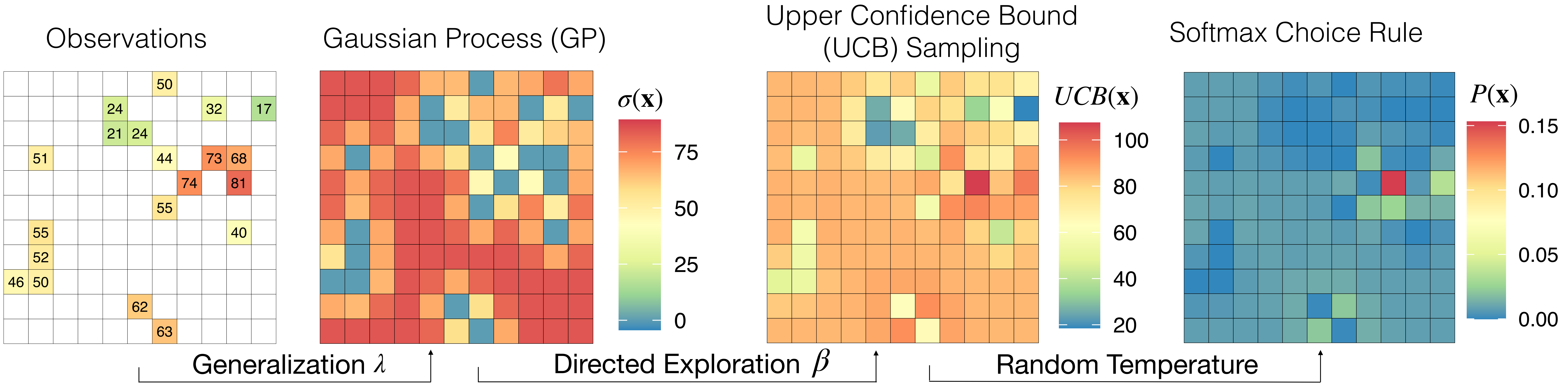


$$k_{RBF}(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{||\mathbf{x}-\mathbf{x}'||^2}{2\lambda^2}\right) \qquad UCB(\mathbf{x}) = \mu(\mathbf{x}) + \beta\sigma(\mathbf{x})$$





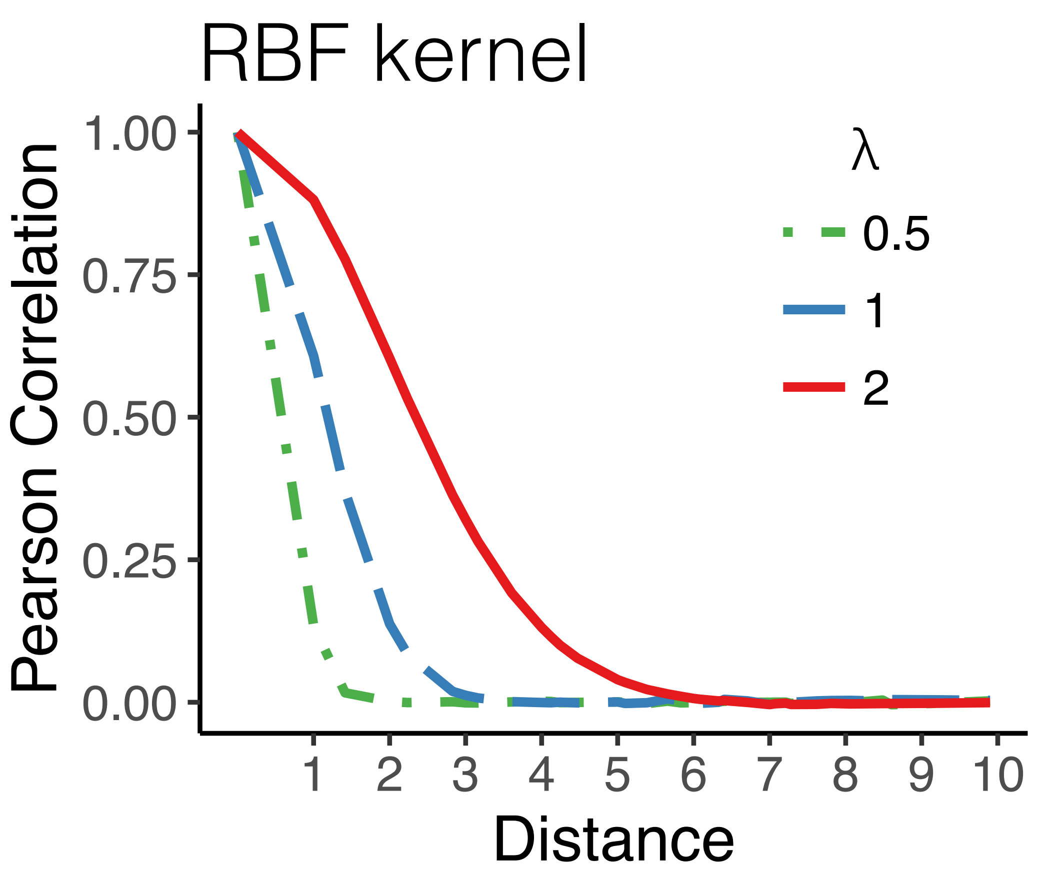
GP-UCB Model

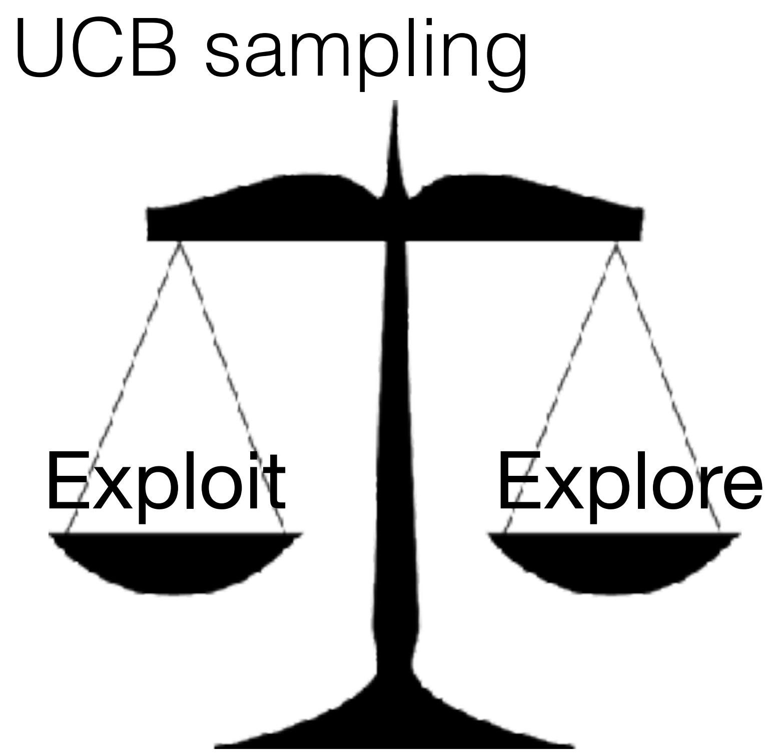


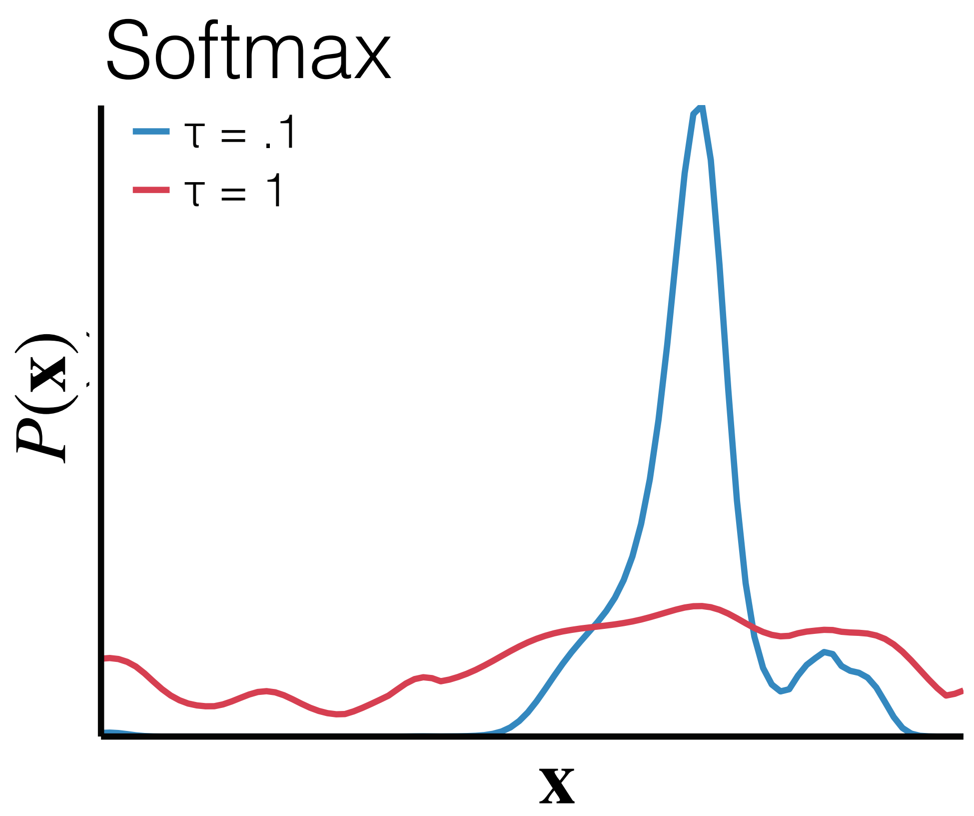
$$k_{RBF}(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{||\mathbf{x} - \mathbf{x}'||^2}{2\lambda^2}\right)$$

$$UCB(\mathbf{x}) = \mu(\mathbf{x}) + \beta\sigma(\mathbf{x})$$

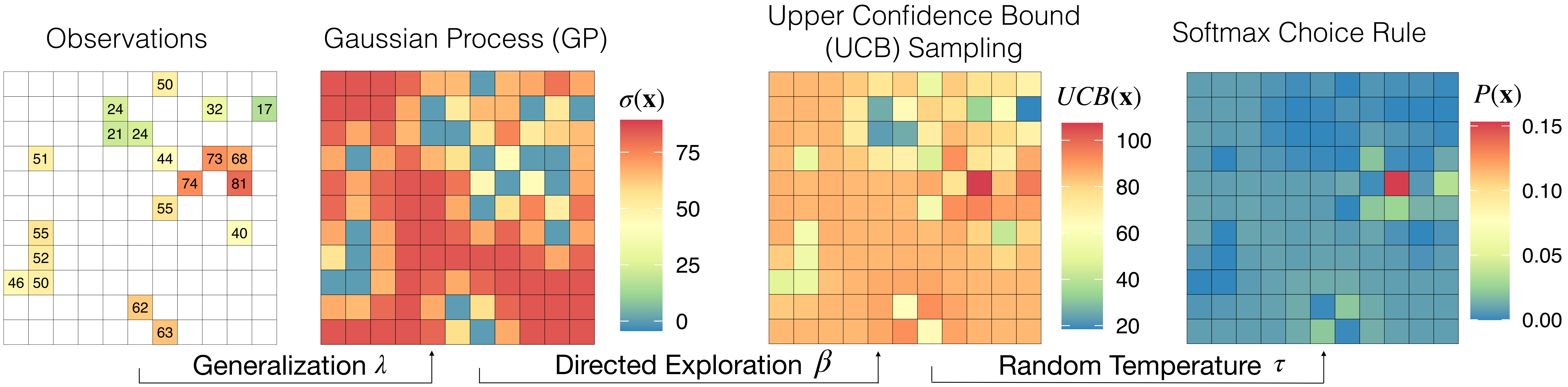
$$P(\mathbf{x}) \propto \exp(UCB(\mathbf{x})/\tau)$$







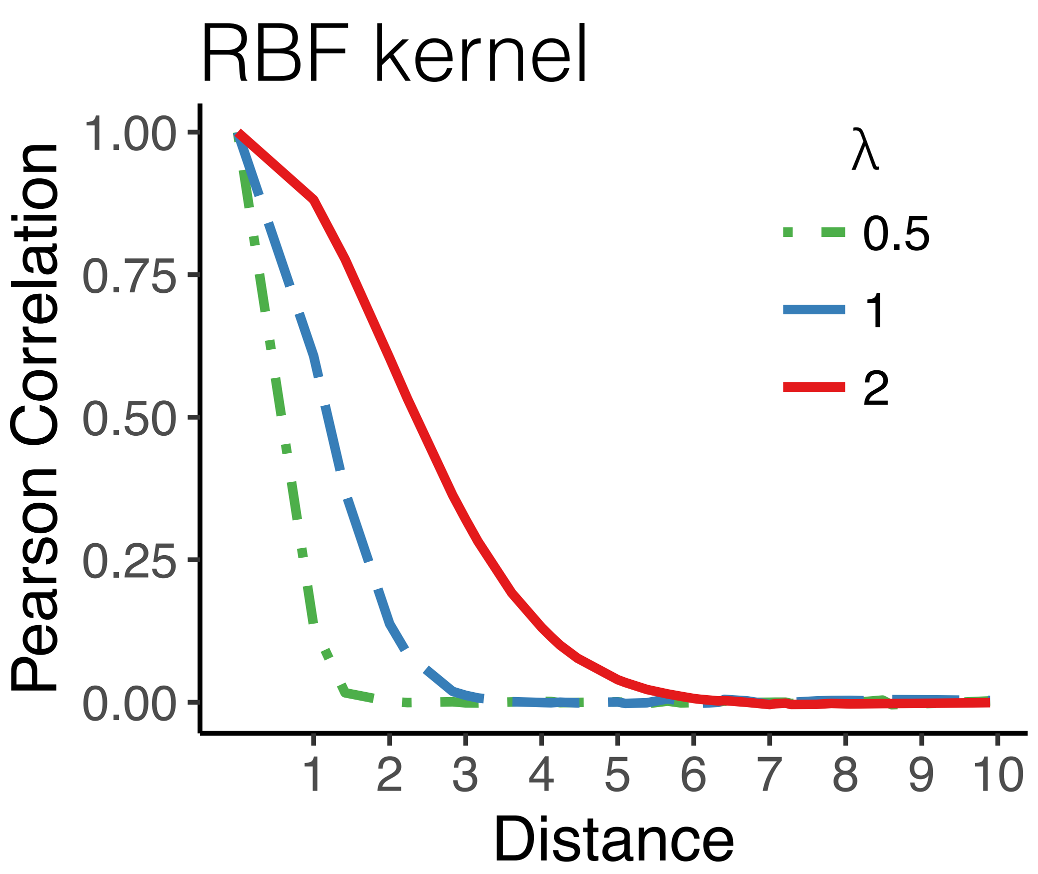
GP-UCB Model

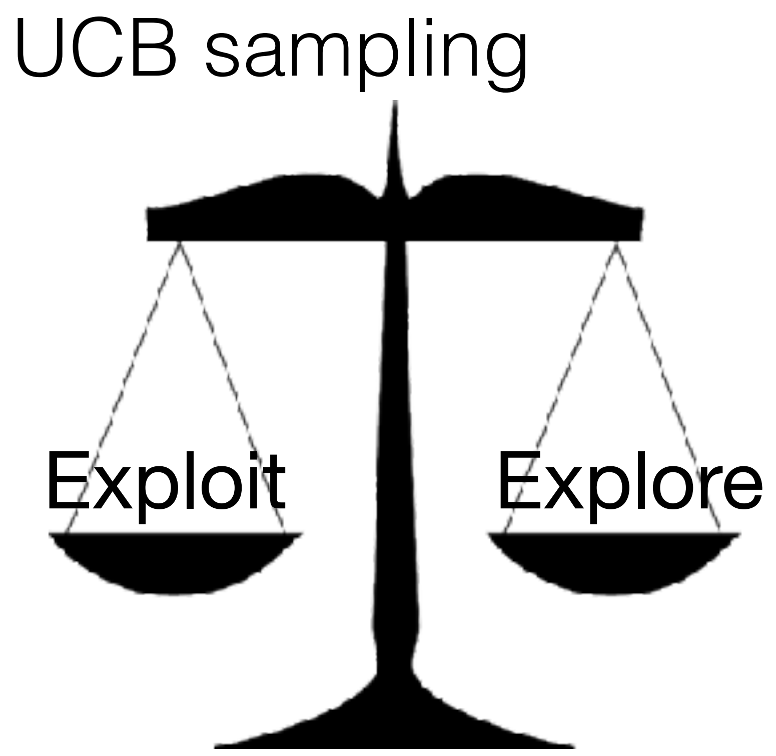


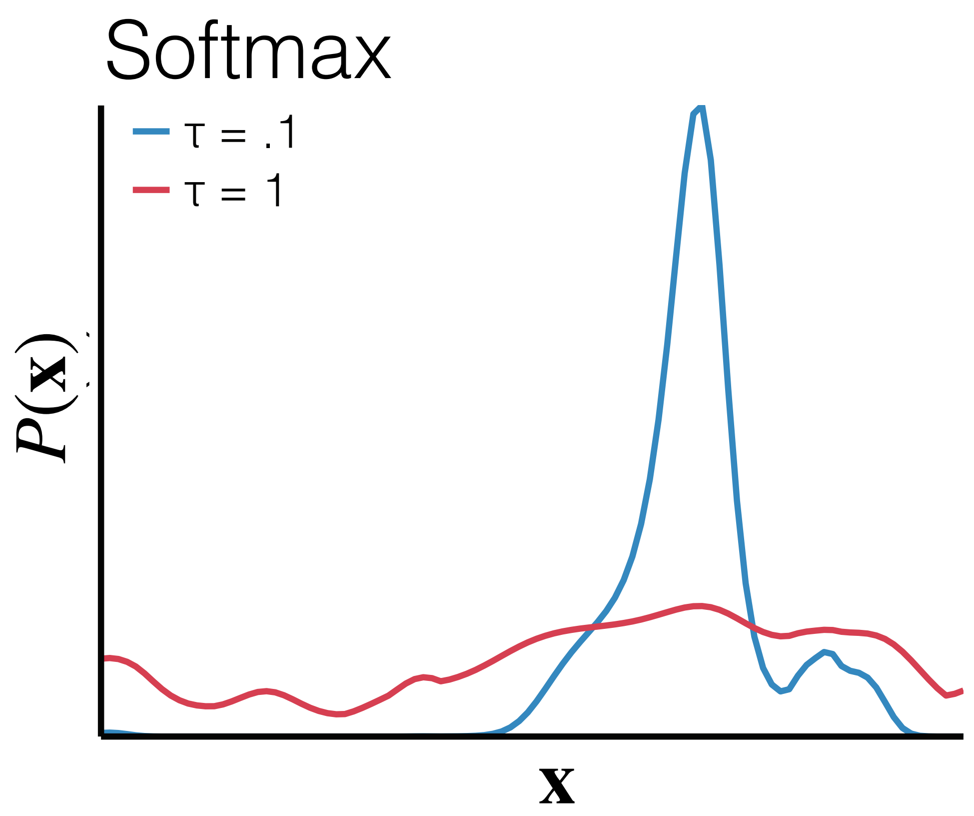
$$k_{RBF}(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{||\mathbf{x}-\mathbf{x}'||^2}{2\lambda^2}\right)$$

$$UCB(\mathbf{x}) = \mu(\mathbf{x}) + \beta\sigma(\mathbf{x})$$

$$P(\mathbf{x}) \propto \exp(UCB(\mathbf{x})/\tau)$$







Anna Giron
Uni Tübingen

Simon Ciranka
MPI Berlin

nature human behaviour

Article

https://doi.org/10.1038/s41562-023-01662-1

Developmental changes in exploration resemble stochastic optimization

Received: 11 November 2022

Accepted: 21 June 2023

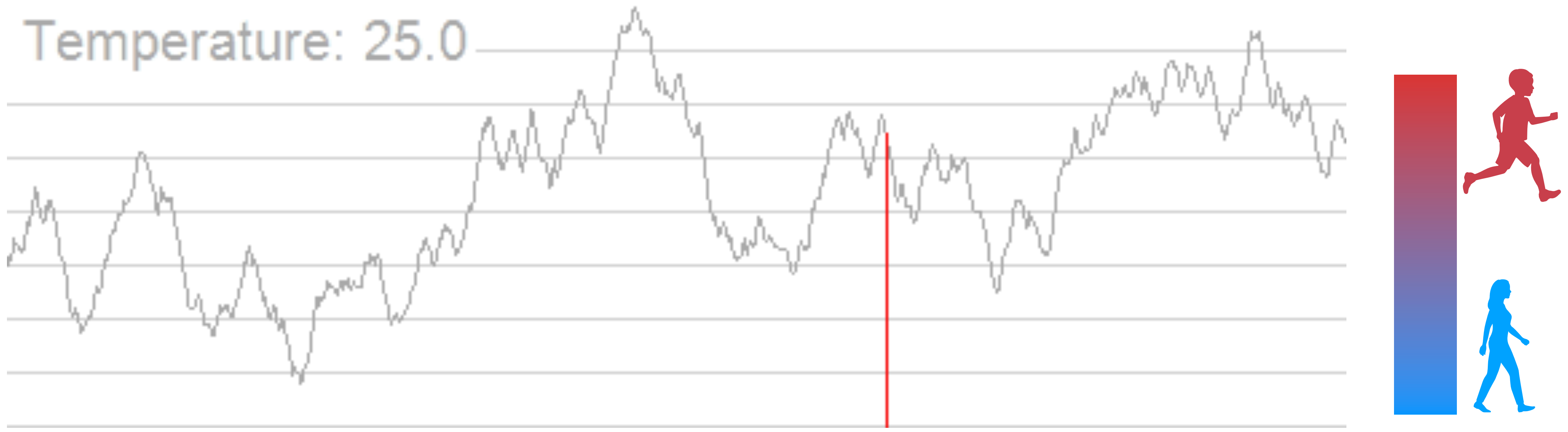
Anna P. Giron[1,2,12], Simon Ciranka[3,4,12], Eric Schulz[5], Wouter van den Bos[6,7], Azzurra Ruggeri[8,9,10], Björn Meder[8,11] & Charley M. Wu[1,3]

Gopnik et al., (*PNAS* 2017)

Development as “cooling off”

- **Inspiration**: Heated metal becomes less malleable as it cools

Stochastic Optimization

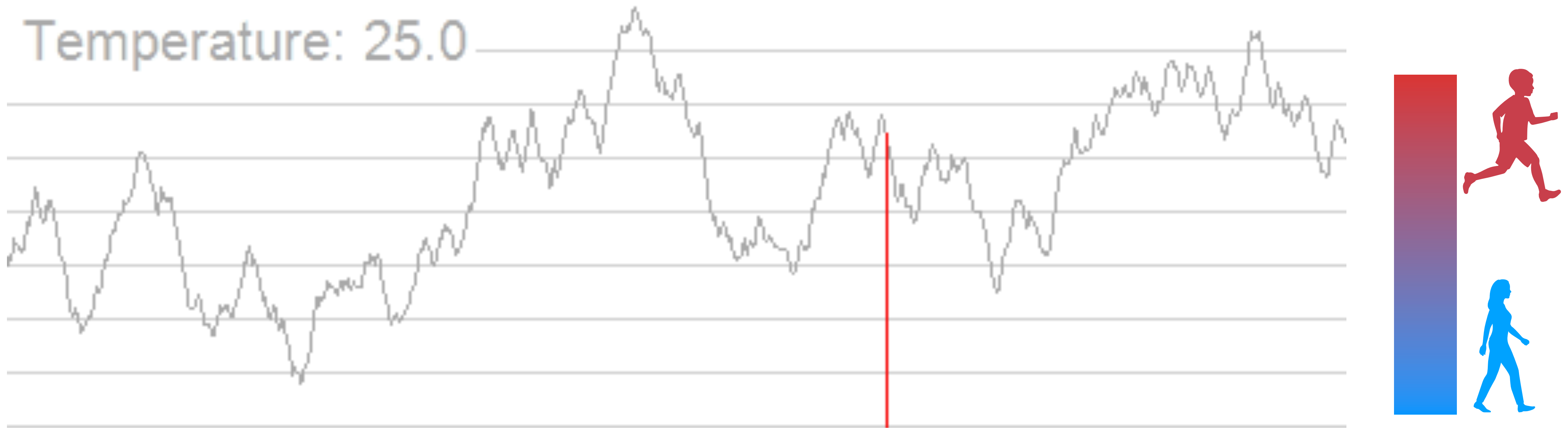


Gopnik et al., (*PNAS* 2017)

Development as "cooling off"

- **Inspiration**: Heated metal becomes less malleable as it cools
- **Application**:
  - Optimization algorithms start off very explorative (high temperature) and gradually becomes more exploitative (cools off)
  - Avoids getting stuck in a local optima

Stochastic Optimization

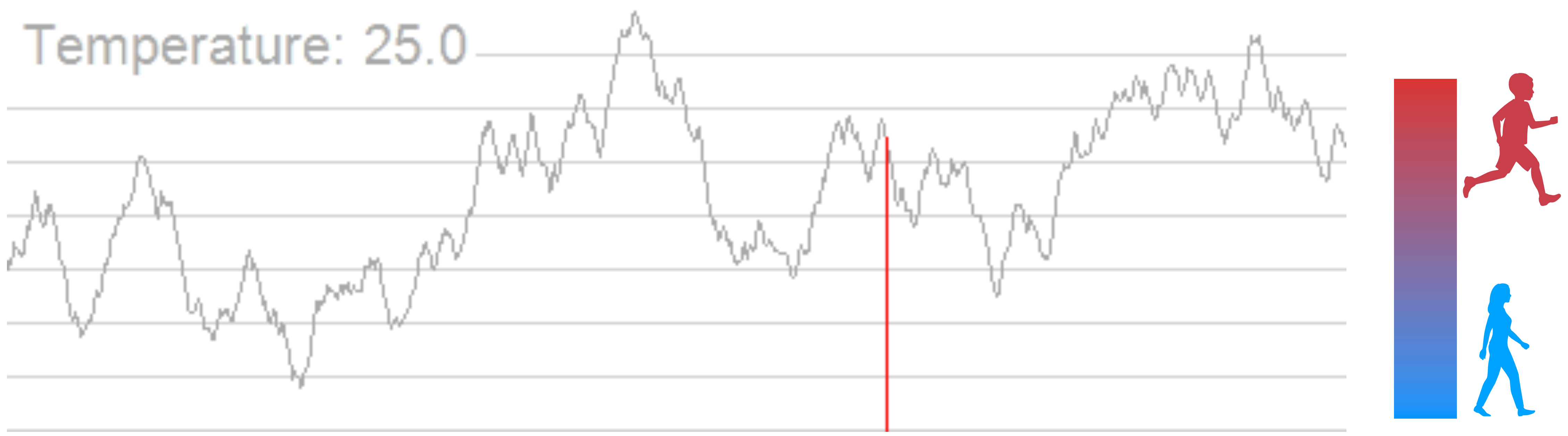


Gopnik et al., (*PNAS* 2017)

Development as "cooling off"

- **Inspiration**: Heated metal becomes less malleable as it cools
- **Application**:
    - Optimization algorithms start off very explorative (high temperature) and gradually becomes more exploitative (cools off)
    - Avoids getting stuck in a local optima
- **Theory of development:**
    - "Cooling off" as an explanation for high variability of children's decisions/hypotheses

Stochastic Optimization

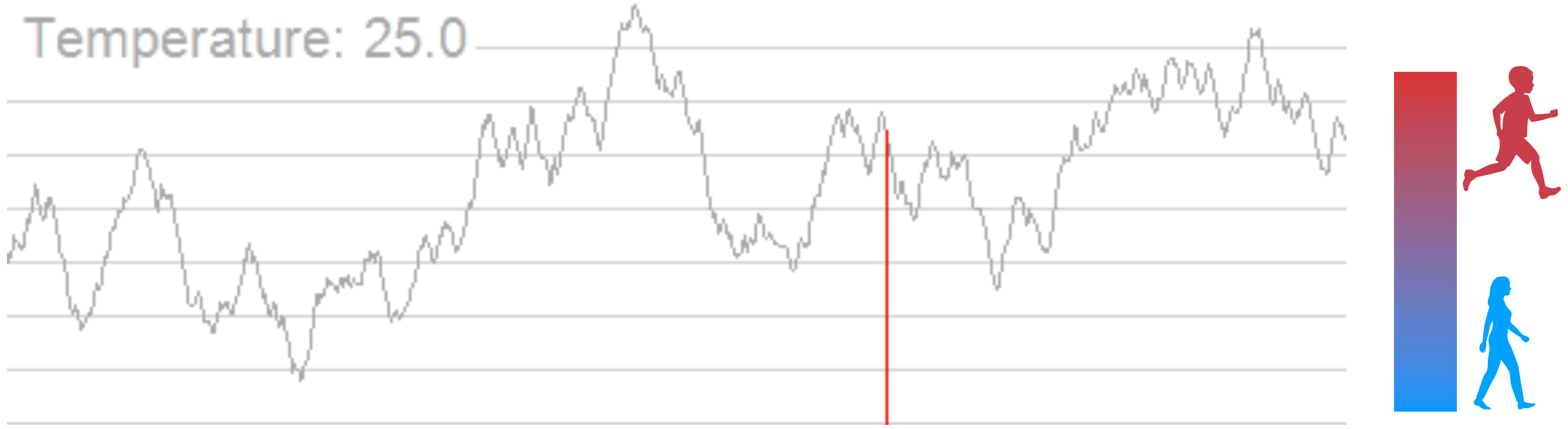


Gopnik et al., (*PNAS* 2017)

Development as "cooling off"

- **Inspiration**: Heated metal becomes less malleable as it cools
- **Application**:
  - Optimization algorithms start off very explorative (high temperature) and gradually becomes more exploitative (cools off)
  - Avoids getting stuck in a local optima
- **Theory of development:**
  - "Cooling off" as an explanation for high variability of children's decisions/hypotheses
- **Implementation: ?**
  - Lack of a direct empirical test
  - Ambuigity in what is being optimized

Stochastic Optimization

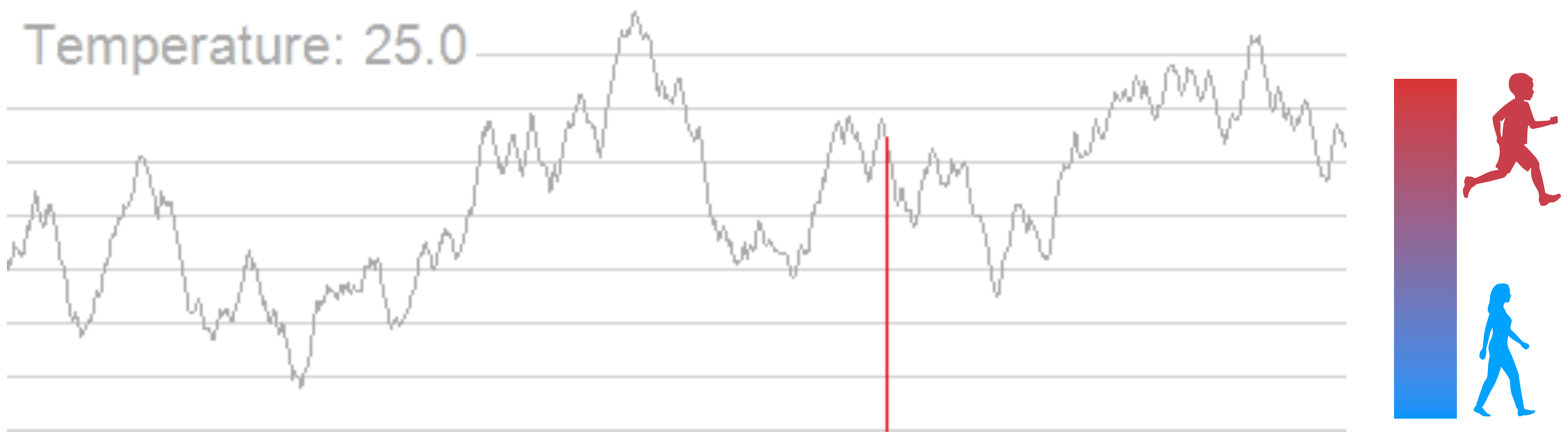


Gopnik et al., (*PNAS* 2017)

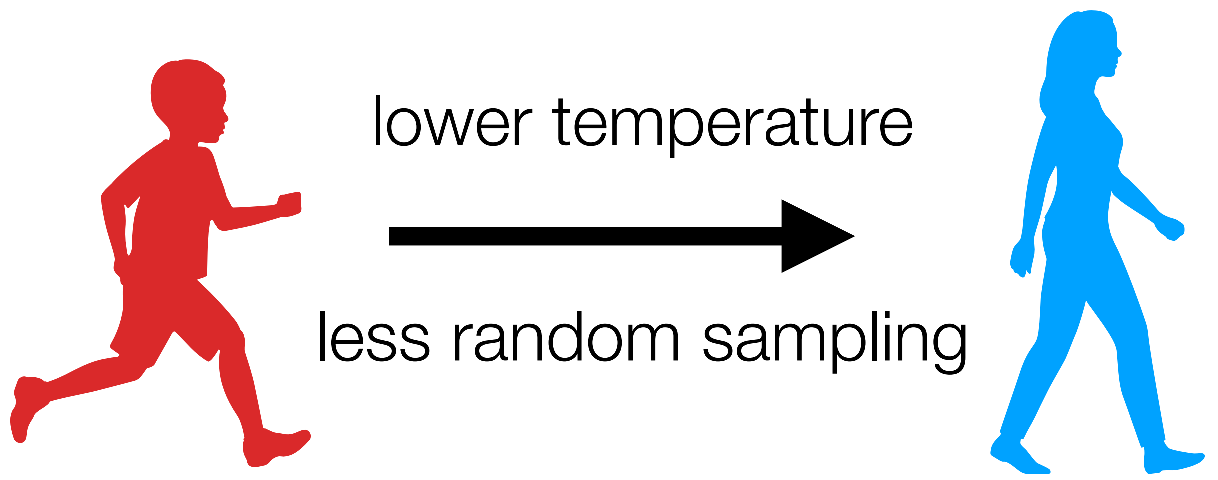
Development as “cooling off”

- **Inspiration**: Heated metal becomes less malleable as it cools
- **Application**:
  - Optimization algorithms start off very explorative (high temperature) and gradually becomes more exploitative (cools off)
  - Avoids getting stuck in a local optima
- **Theory of development:**
  - “Cooling off” as an explanation for high variability of children’s decisions/hypotheses
- **Implementation: ?**
  - Lack of a direct empirical test
  - Ambuigity in what is being optimized

Stochastic Optimization



H1: Uni-dimensional reduction of randomness in sampling

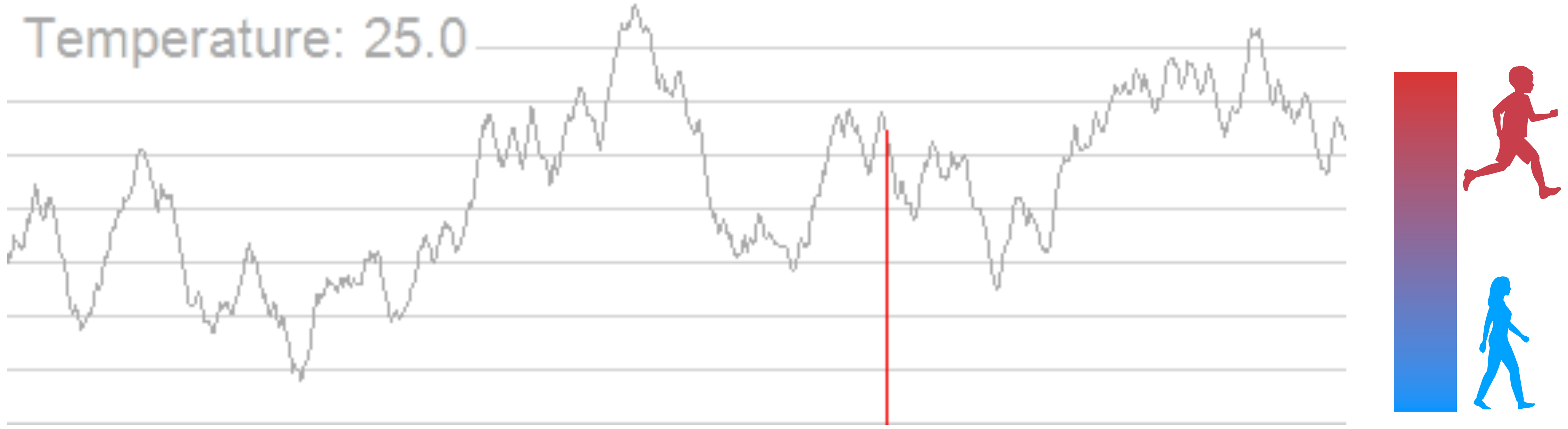


Gopnik et al., (*PNAS* 2017)

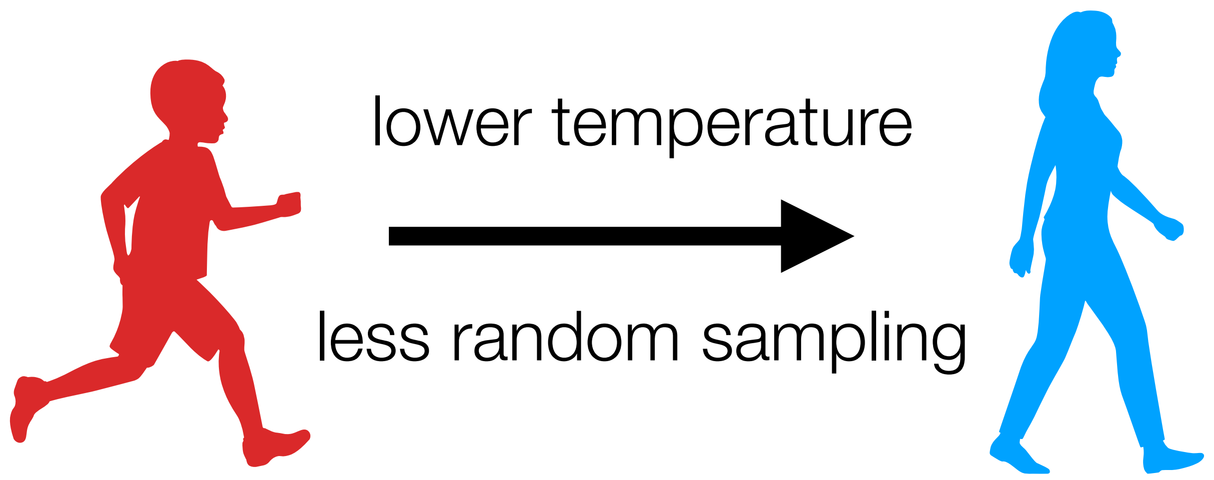
Development as "cooling off"

- **Inspiration**: Heated metal becomes less malleable as it cools
- **Application**:
  - Optimization algorithms start off very explorative (high temperature) and gradually becomes more exploitative (cools off)
  - Avoids getting stuck in a local optima
- **Theory of development:**
  - "Cooling off" as an explanation for high variability of children's decisions/hypotheses
- **Implementation: ?**
  - Lack of a direct empirical test
  - Ambuigity in what is being optimized

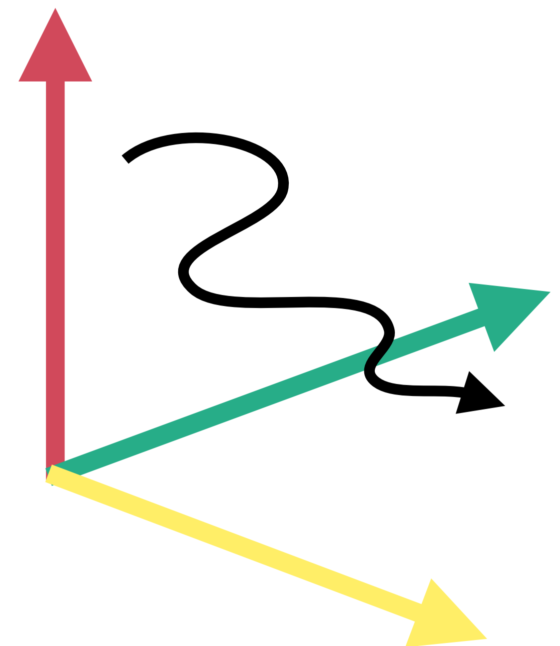
Stochastic Optimization



H1: Uni-dimensional reduction of randomness in sampling

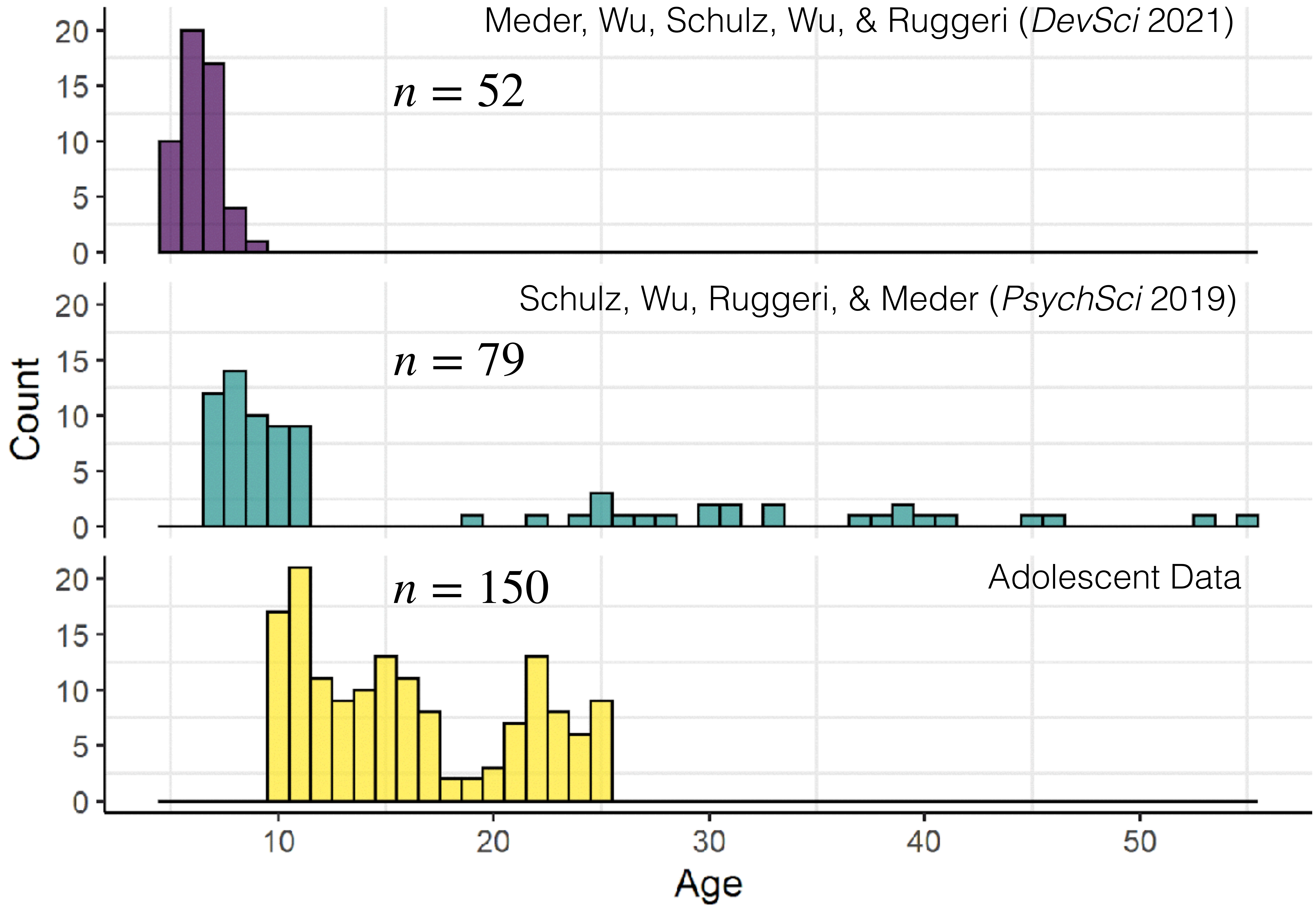


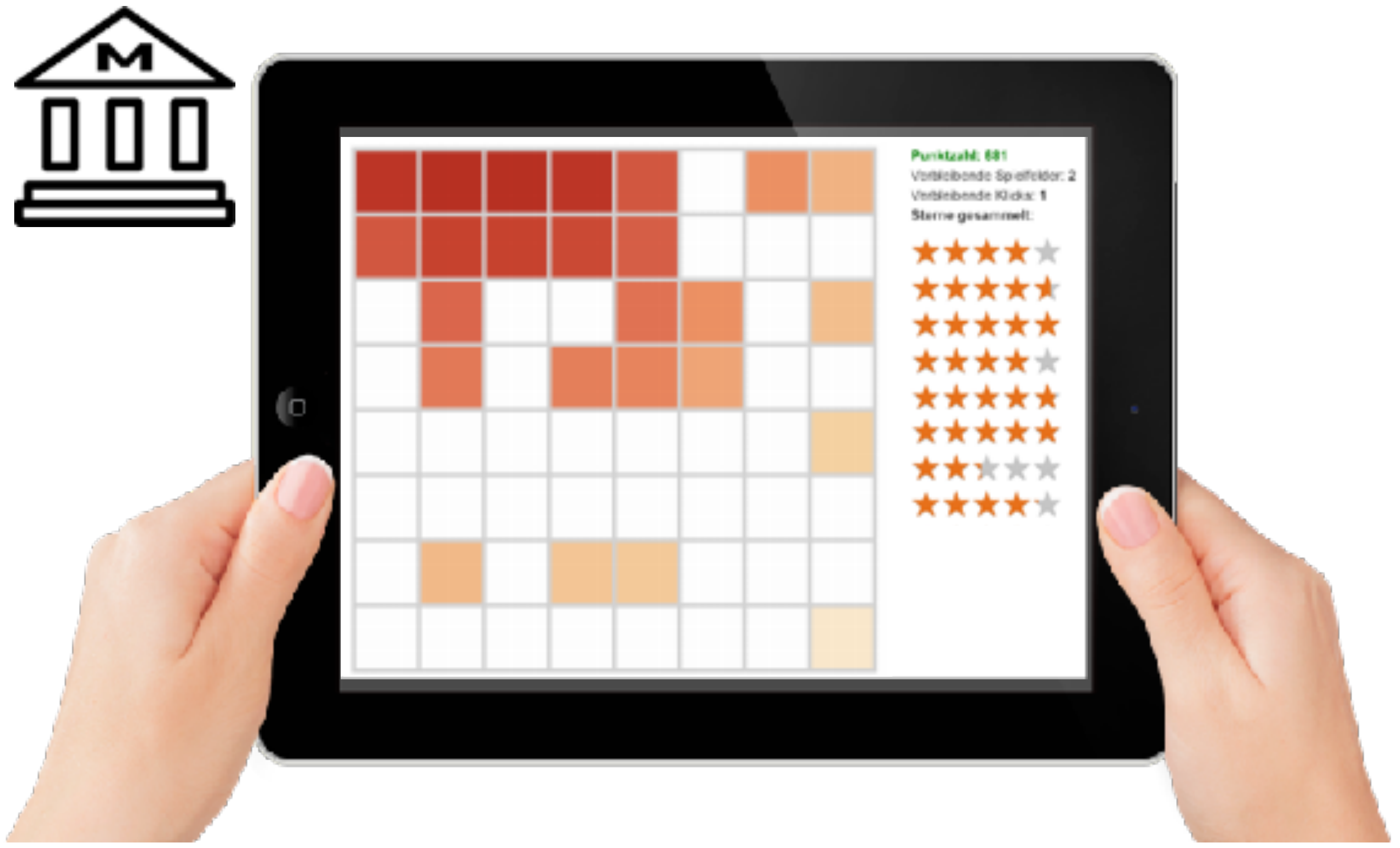
H2: Multi-dimensional optimization of learning strategies

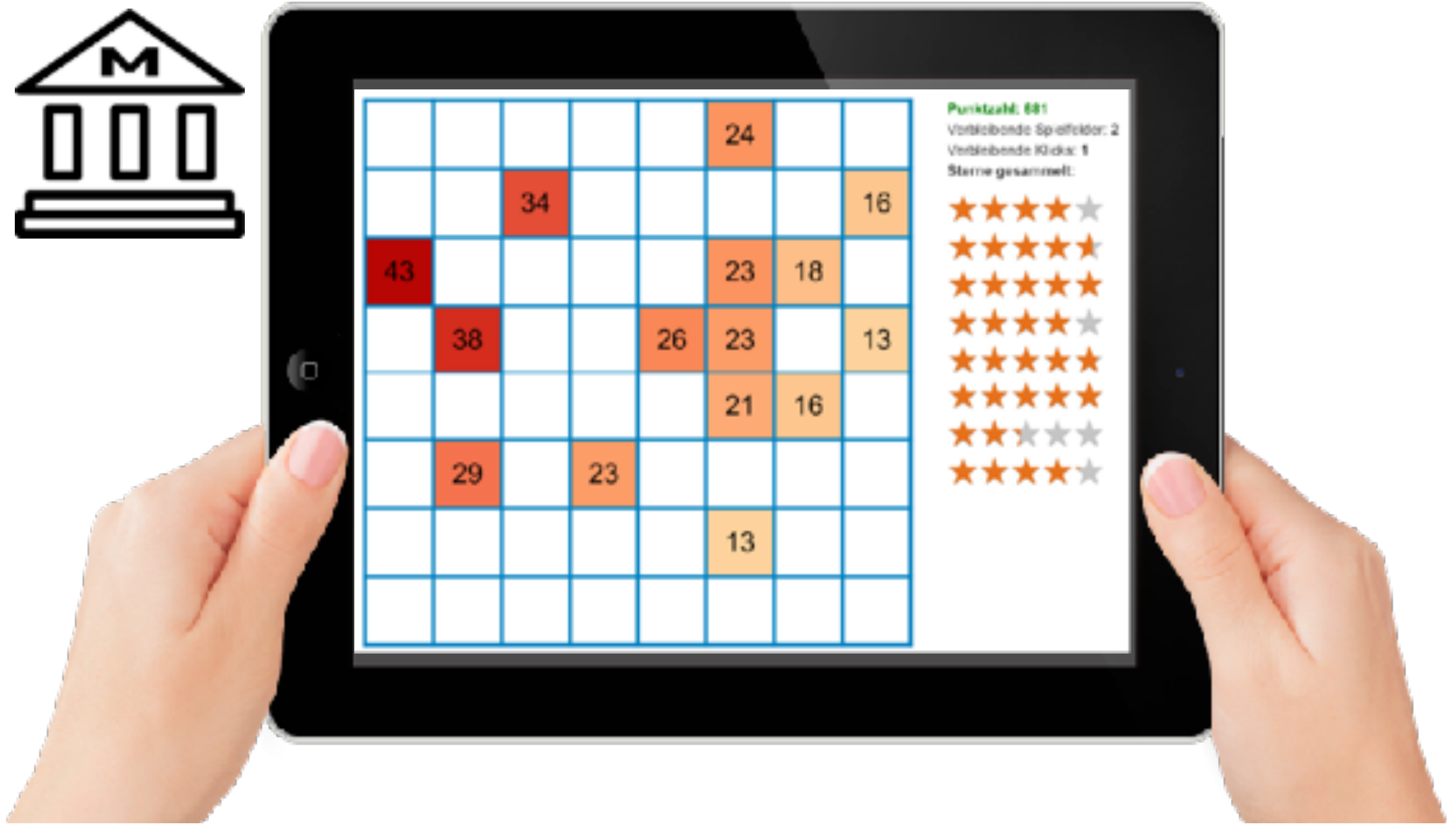


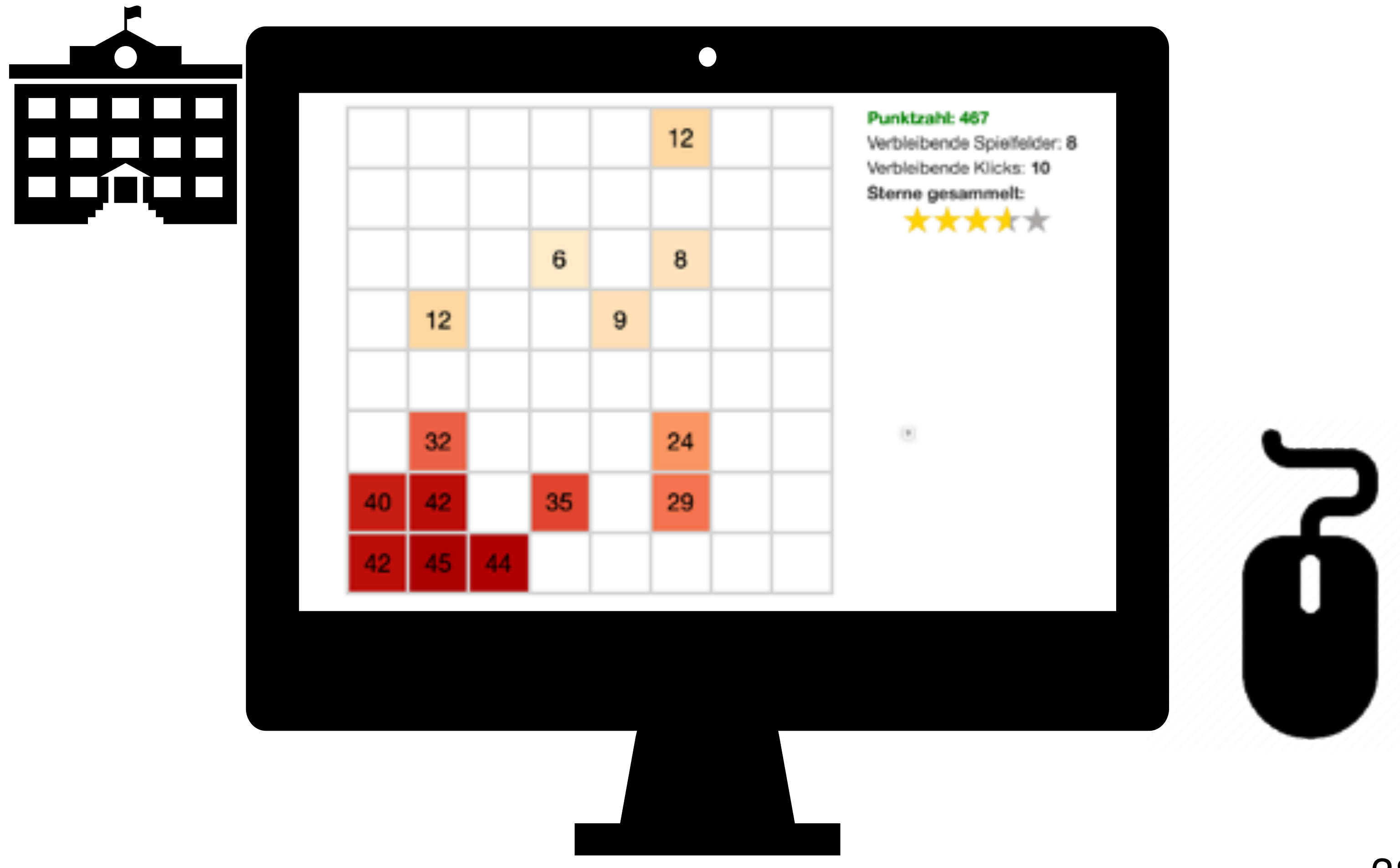
Giron*, Ciranka*, Schulz, van den Bos, Ruggeri, Meder, & Wu (*NHB* 2023)

Combined dataset with *n* = 281 subjects between 5 and 55







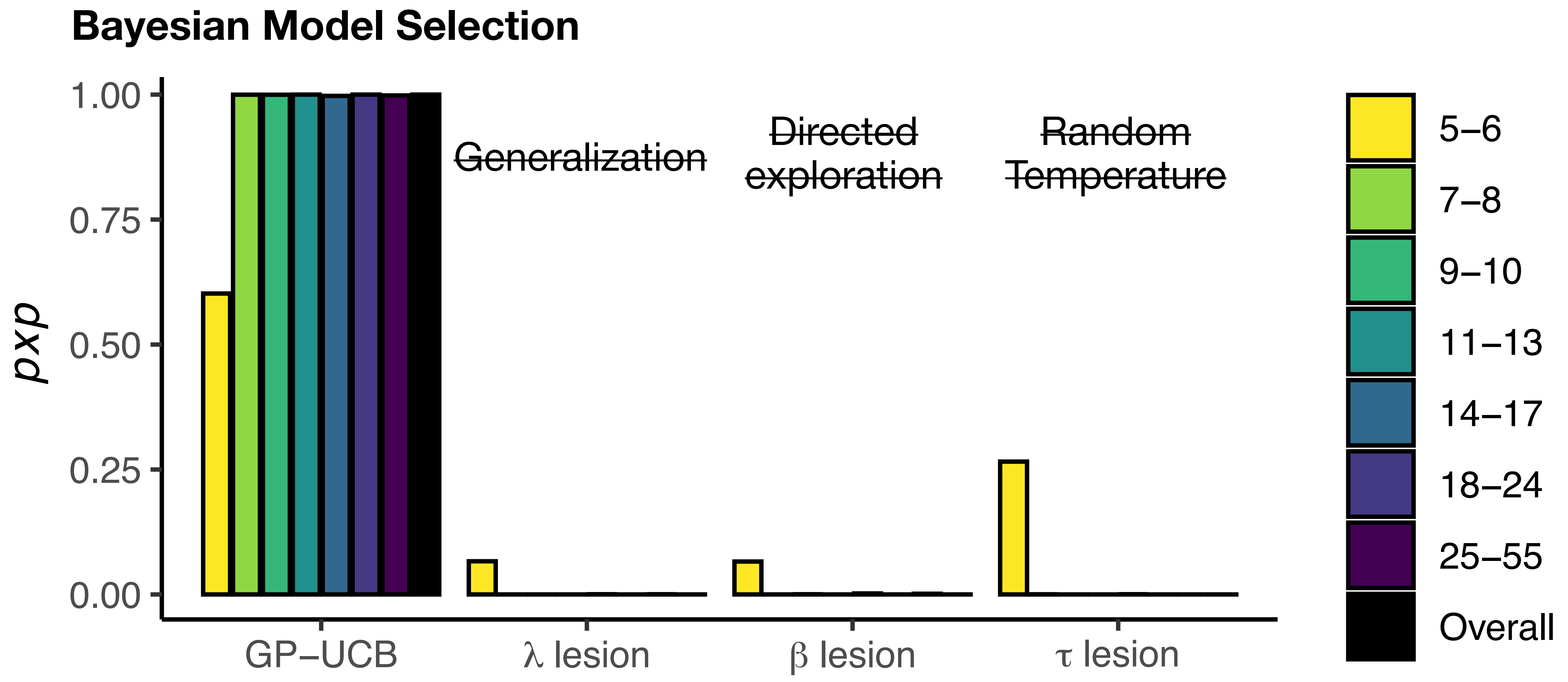


GP-UCB across the lifespan

- GP-UCB provides the predictions of behavior from the ages of 5 to 55 (*n*=281)

GP-UCB across the lifespan

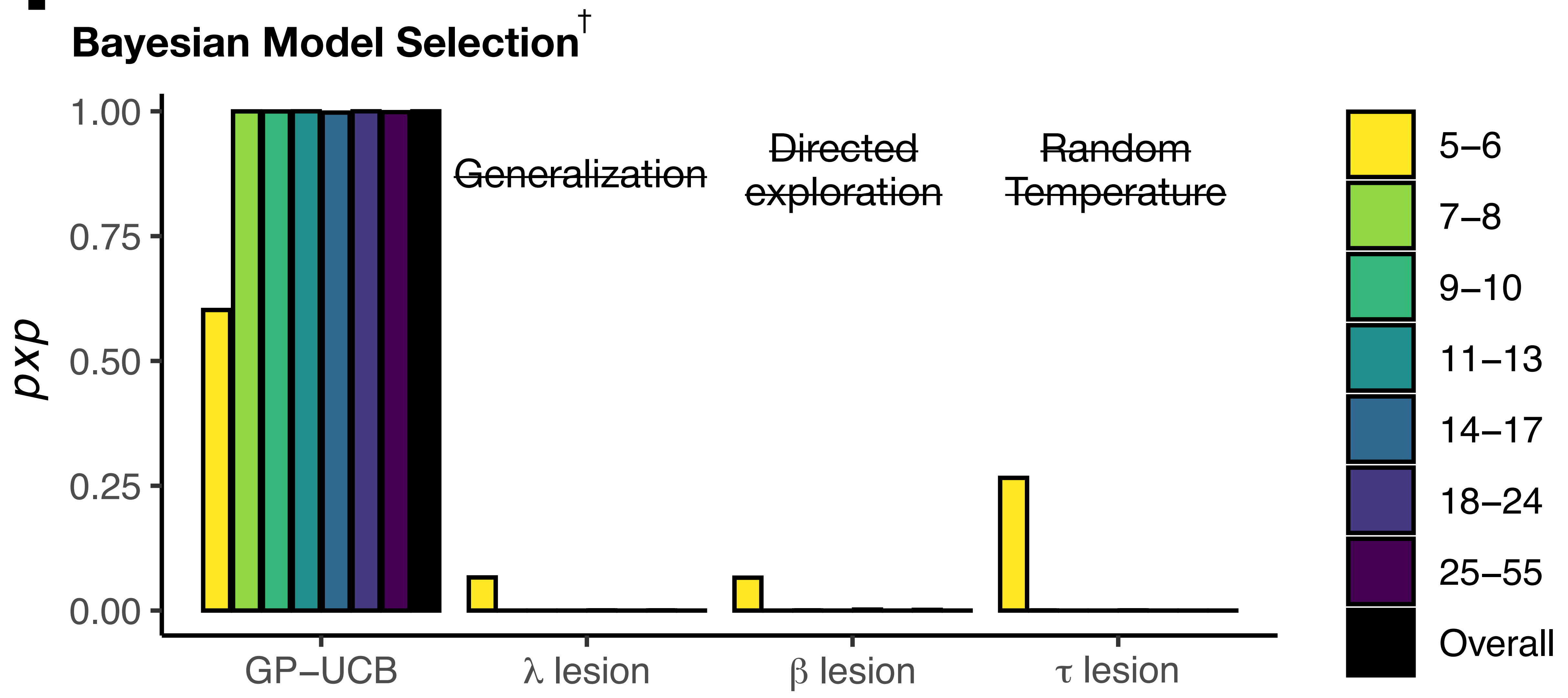
- GP-UCB provides the predictions of behavior from the ages of 5 to 55 (*n*=281)
- We can lesion out each component to show that all are necessary
  - λ lesion replaces GP with a Tabular RL model (i.e., Kalman filter) that learns the value of each option independently without generalization
  - β lesion removes uncertainty-directed exploration by setting $\beta = 0$
  - τ lesion swaps softmax for an ϵ-greedy policy



†Protected Exceedance Probability (Rigoux et al., 2014)

GP-UCB across the lifespan

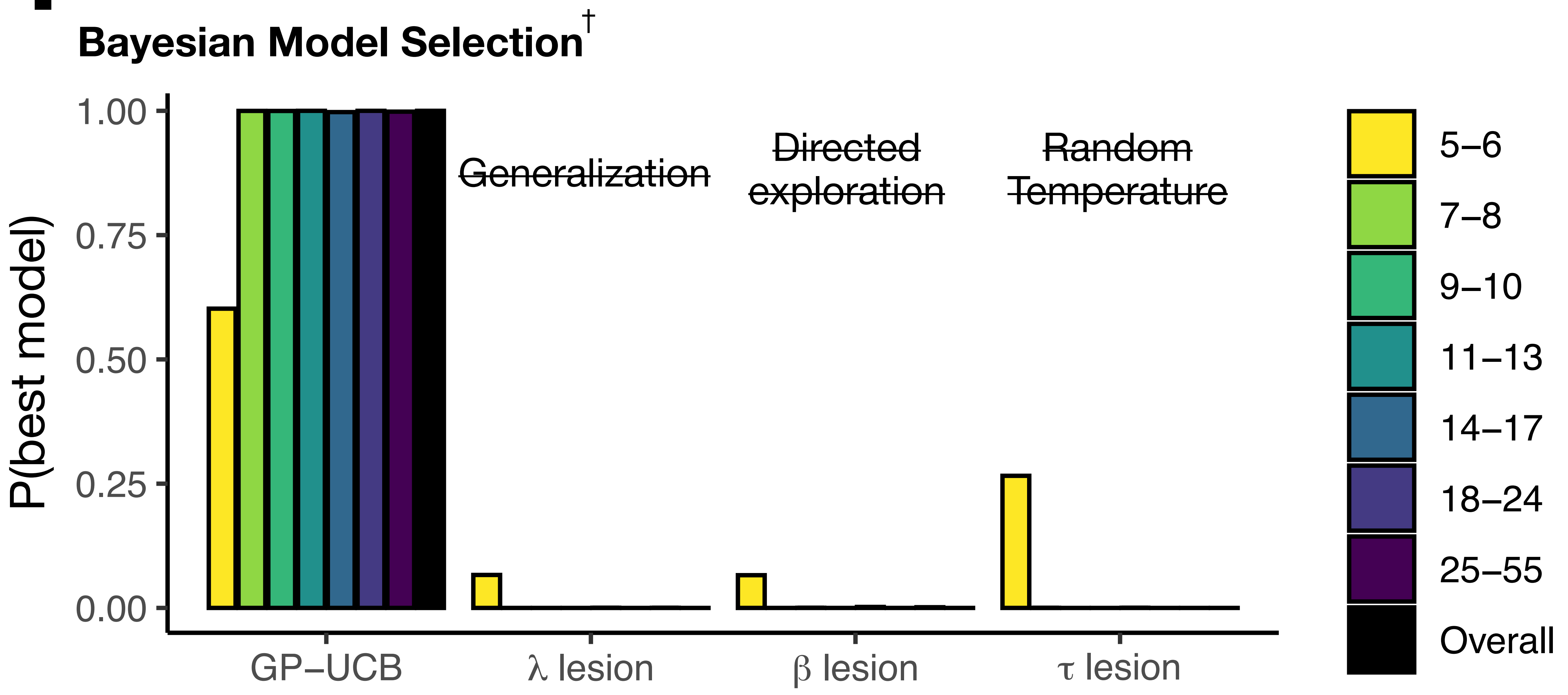
- GP-UCB provides the predictions of behavior from the ages of 5 to 55 (*n*=281)
- We can lesion out each component to show that all are necessary
  - λ lesion replaces GP with a Tabular RL model (i.e., Kalman filter) that learns the value of each option independently without generalization
  - β lesion removes uncertainty-directed exploration by setting $\beta = 0$
  - τ lesion swaps softmax for an ϵ-greedy policy



†Protected Exceedance Probability (Rigoux et al., 2014)

GP-UCB across the lifespan

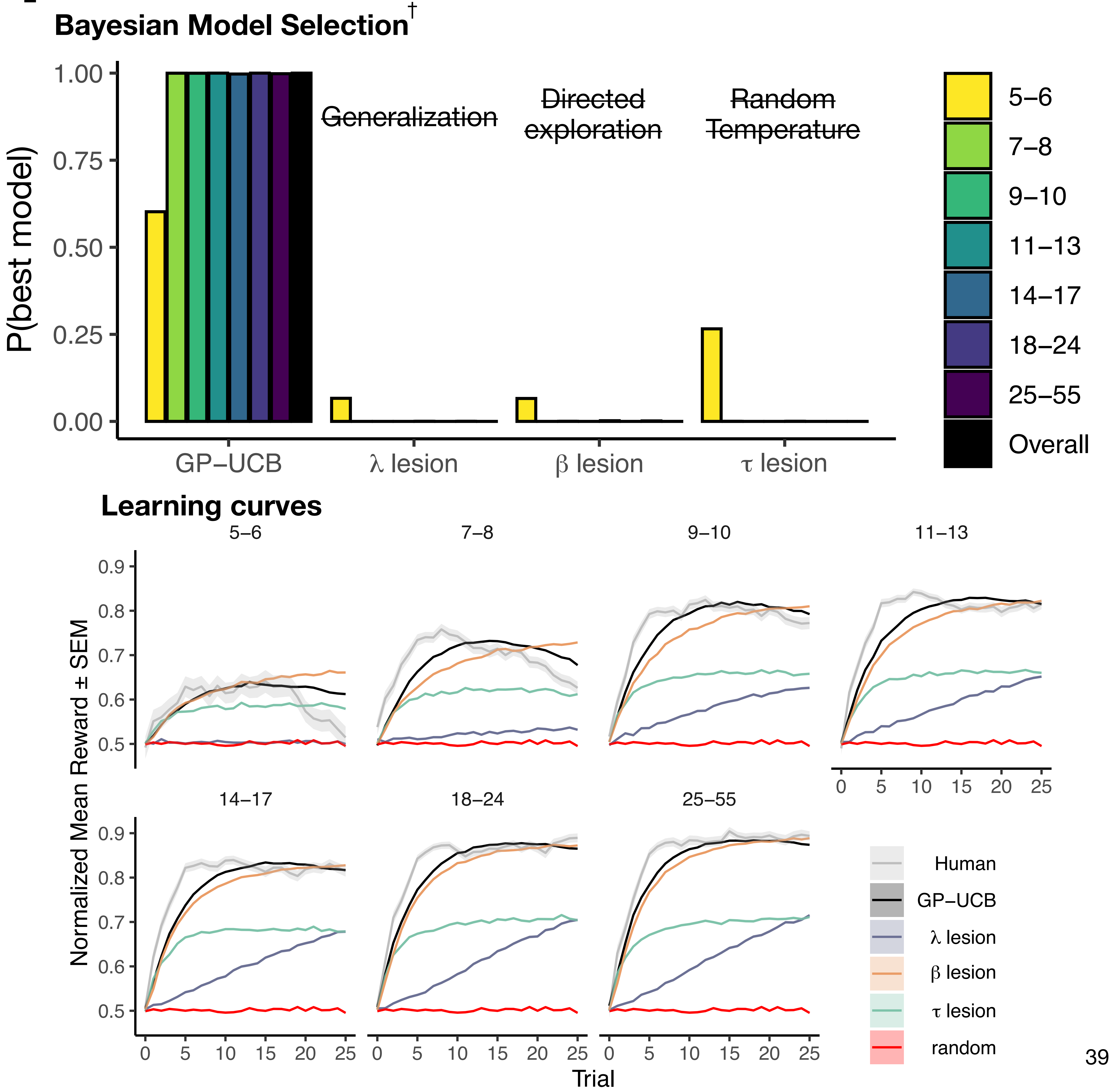
- GP-UCB provides the predictions of behavior from the ages of 5 to 55 (*n*=281)
- We can lesion out each component to show that all are necessary
  - λ lesion replaces GP with a Tabular RL model (i.e., Kalman filter) that learns the value of each option independently without generalization
  - β lesion removes uncertainty-directed exploration by setting $\beta = 0$
  - τ lesion swaps softmax for an ϵ-greedy policy



†Protected Exceedance Probability (Rigoux et al., 2014)

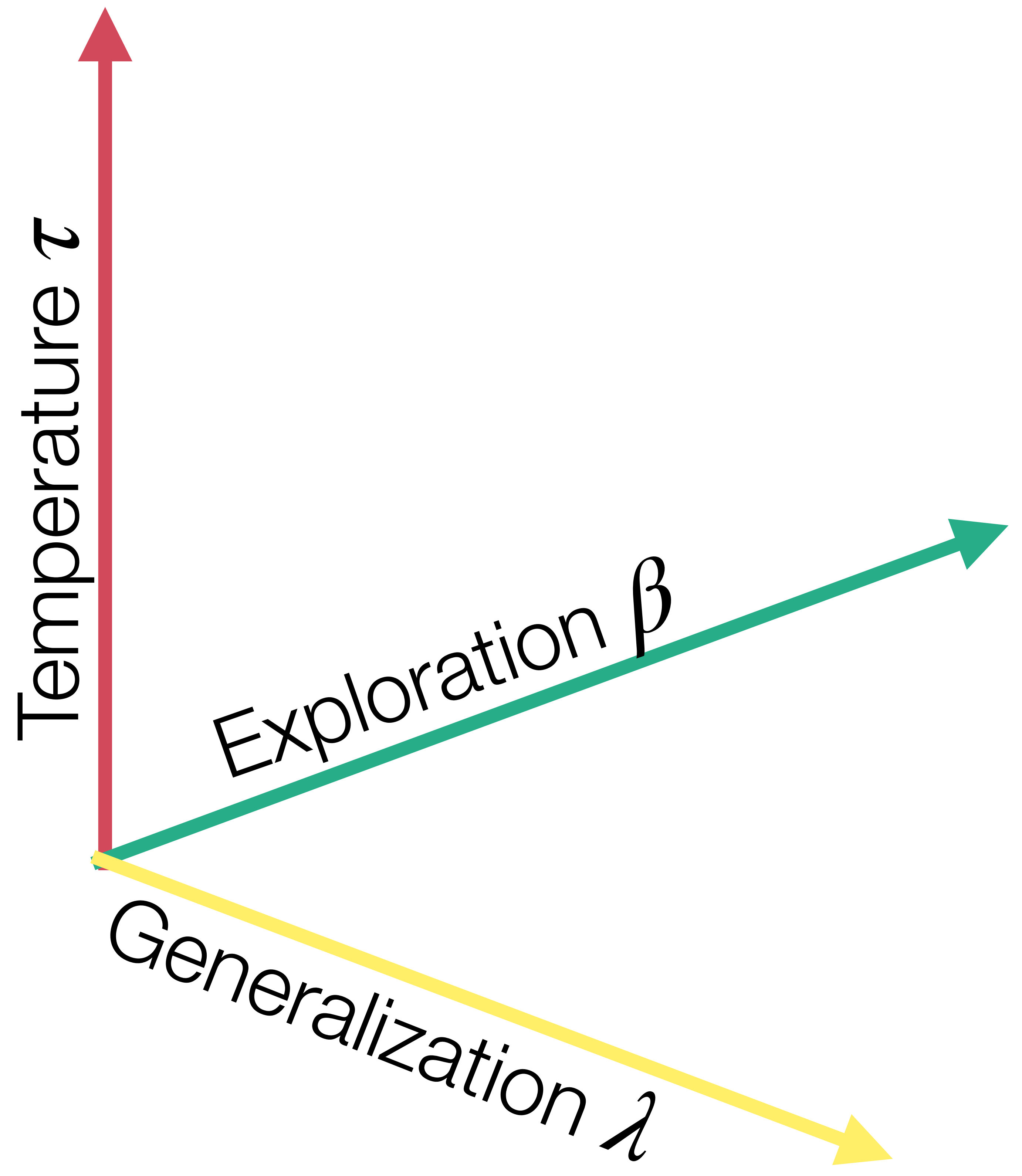
GP-UCB across the lifespan

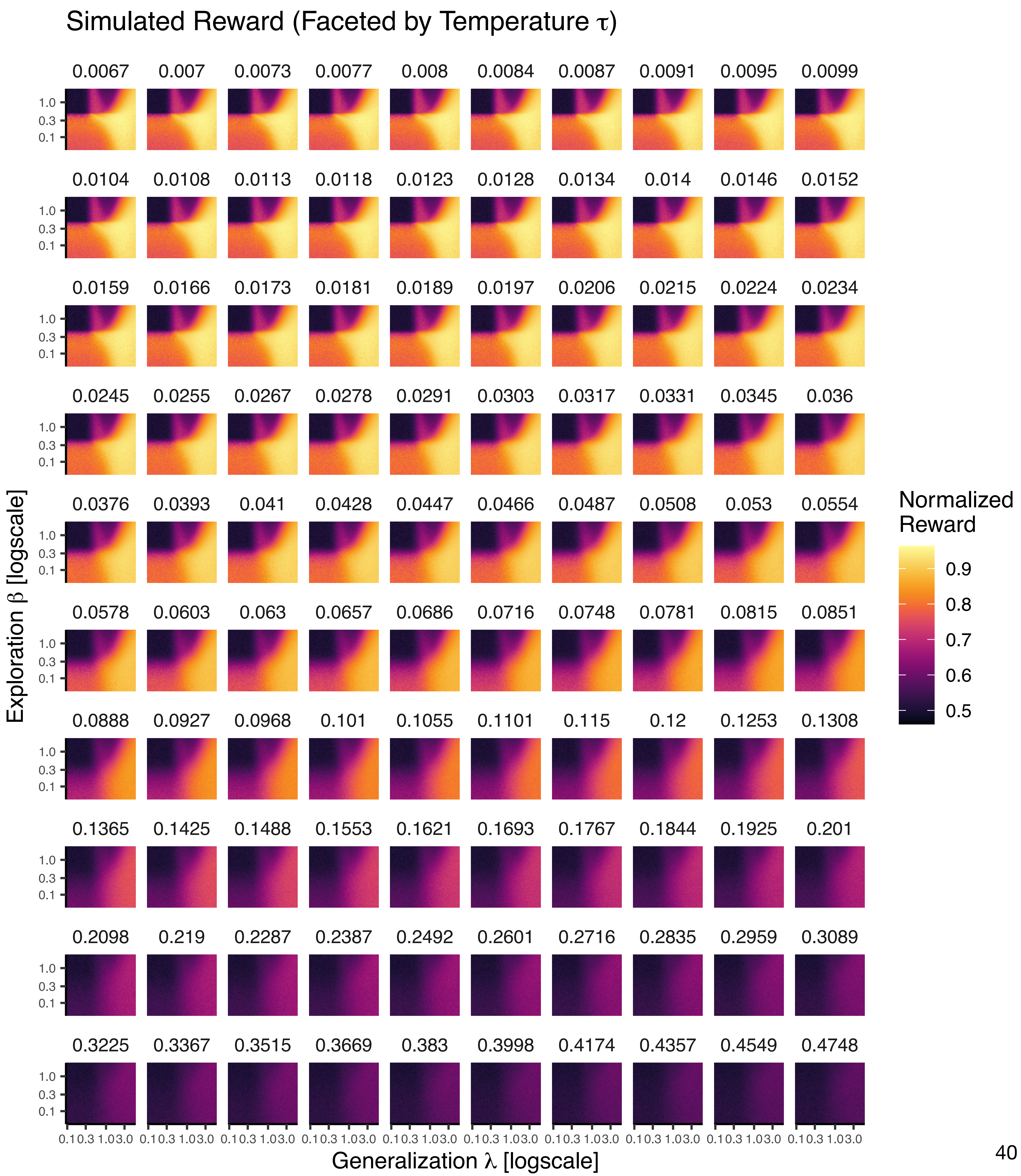
- GP-UCB provides the predictions of behavior from the ages of 5 to 55 (*n*=281)
- We can lesion out each component to show that all are necessary
    - λ lesion replaces GP with a Tabular RL model (i.e., Kalman filter) that learns the value of each option independently without generalization
    - β lesion removes uncertainty-directed exploration by setting $\beta = 0$
    - τ lesion swaps softmax for an ϵ-greedy policy
- The **full model** reproduces the same age-related differences in learning curves
    - β-lesion is also good, but doesn't produce the same decaying learning curves that children have and generally learns slower



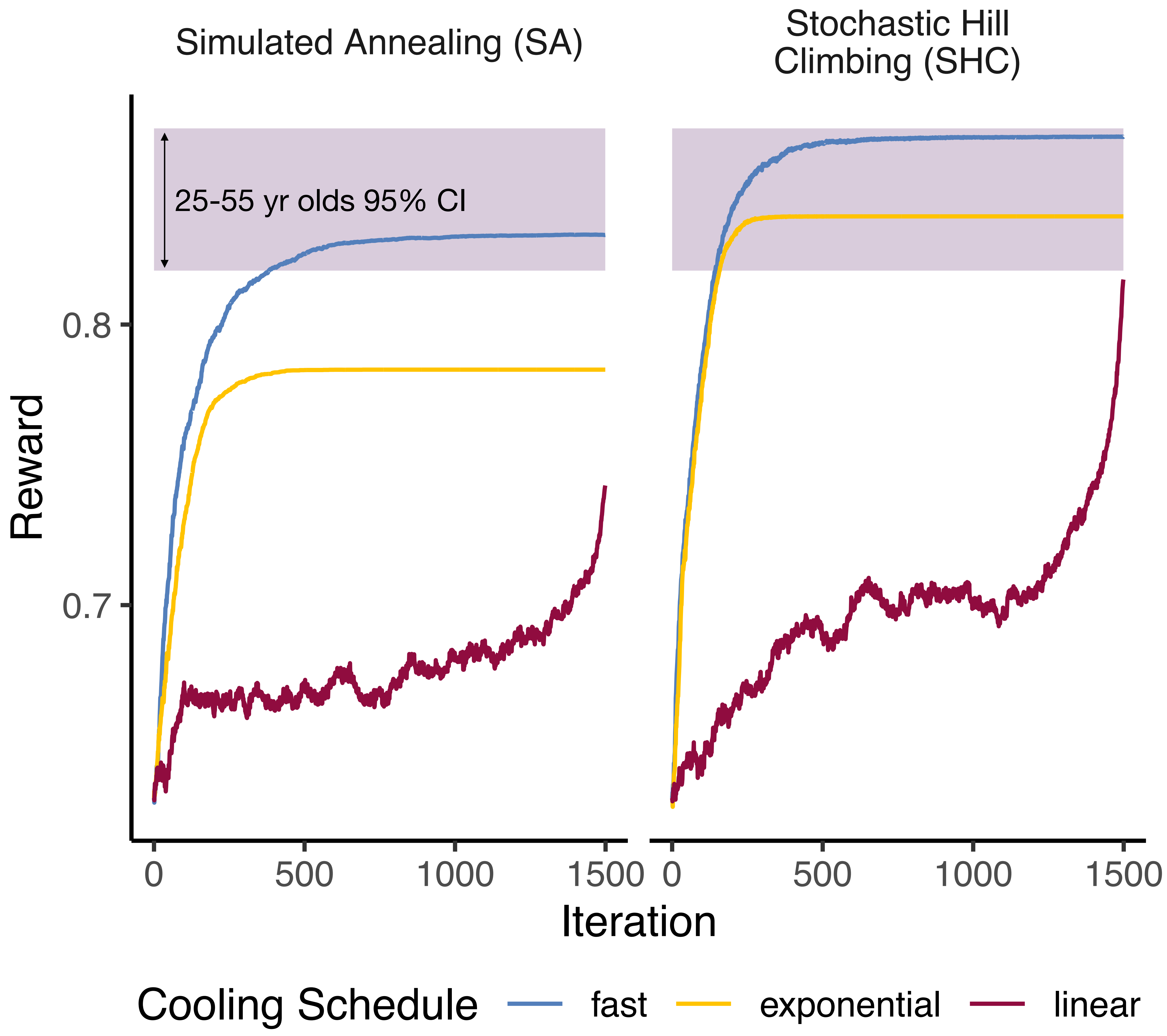
Fitness Landscape

Simulations over 1 million plausible parameter combinations

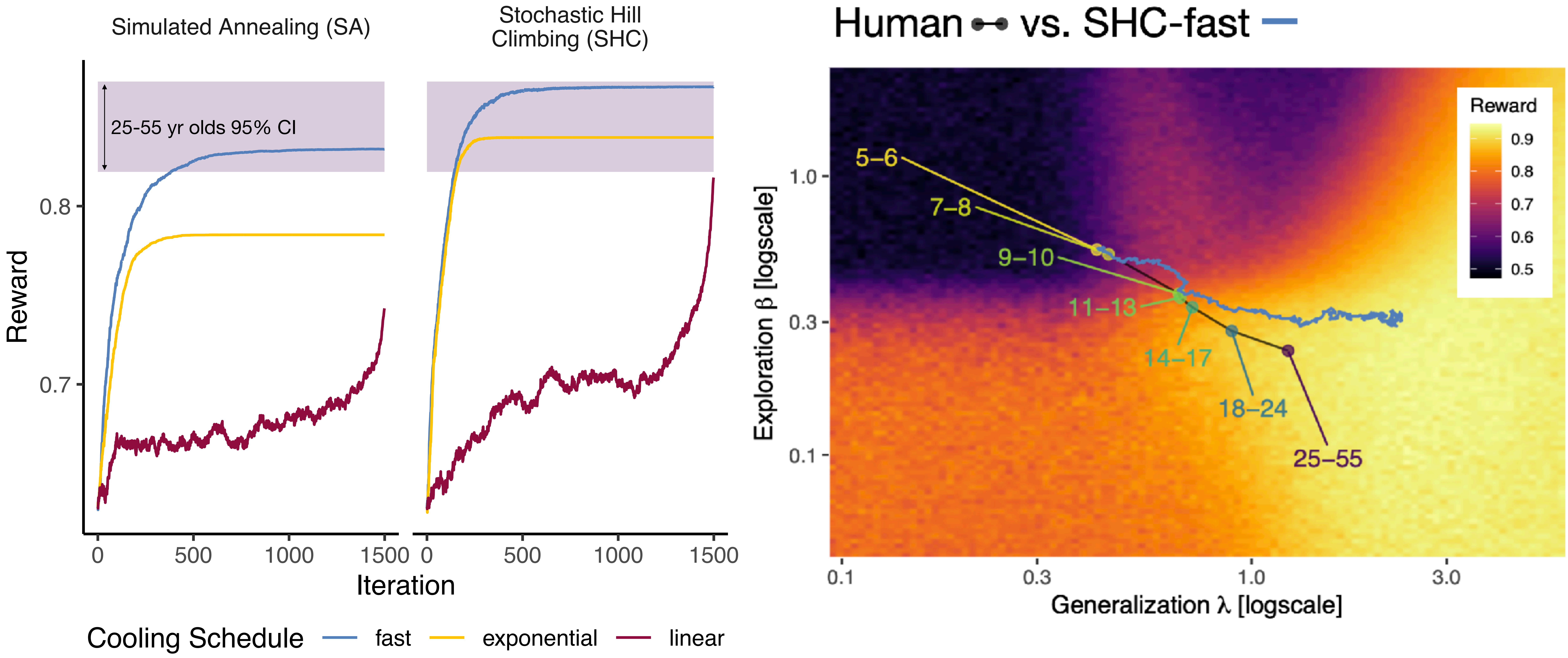




Human development resembles an optimization process in GP parameter space



Human development resembles an optimization process in GP parameter space



A versatile and robust paradigm

OPEN DATA OPEN MATERIALS

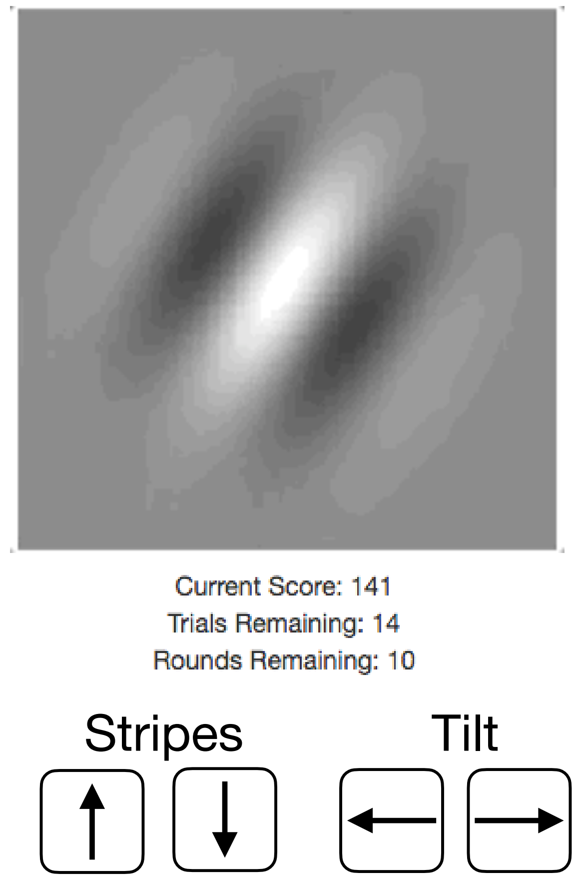
- Generalization guides exploration
  Wu, Schulz, Nelson, Speekenbrink & Meder (*NHB* 2018)

- Developmental trajectory of learning
  Giron*, Ciranka*, Schulz, Van den Bos, Ruggeri, Meder, & Wu (*NHB* 2023)
  Meder, Wu, Schulz & Ruggeri (*DevSci* 2021)
  Schulz, Wu, Ruggeri & Meder (*PsychSci* 2019)

A versatile and robust paradigm

- Generalization guides exploration
  Wu, Schulz, Nelson, Speekenbrink & Meder (*NHB* 2018)
- Developmental trajectory of learning
  Giron*, Ciranka*, Schulz, Van den Bos, Ruggeri, Meder, & Wu (*NHB* 2023)
  Meder, Wu, Schulz & Ruggeri (*DevSci* 2021)
  Schulz, Wu, Ruggeri & Meder (*PsychSci* 2019)
- Search in abstract conceptual spaces
  Wu, Schulz, Garvert, Meder & Schuck (*PLOS Comp Bio, 2020*)

Conceptual features

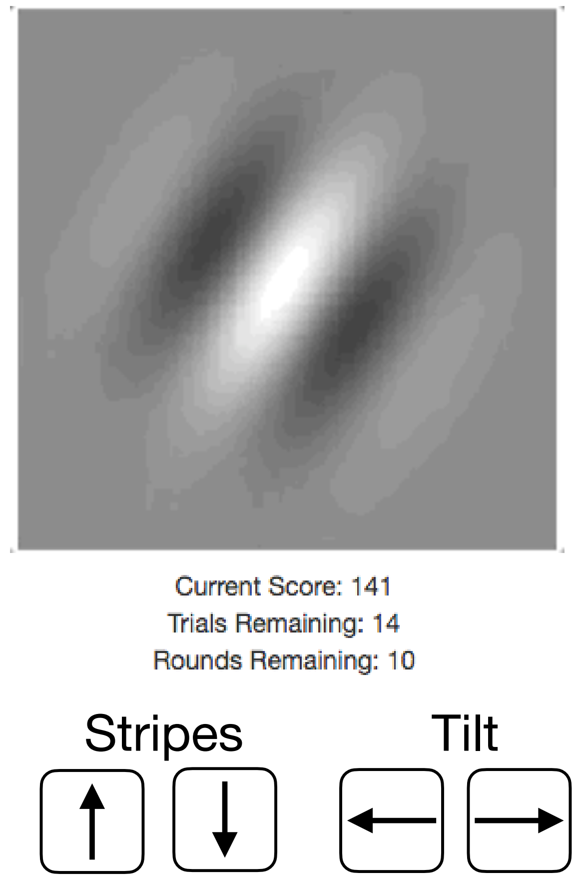


Wu, Schulz, Garvert, Meder & Schuck
(*PLOS Comp Bio 2020*)

A versatile and robust paradigm

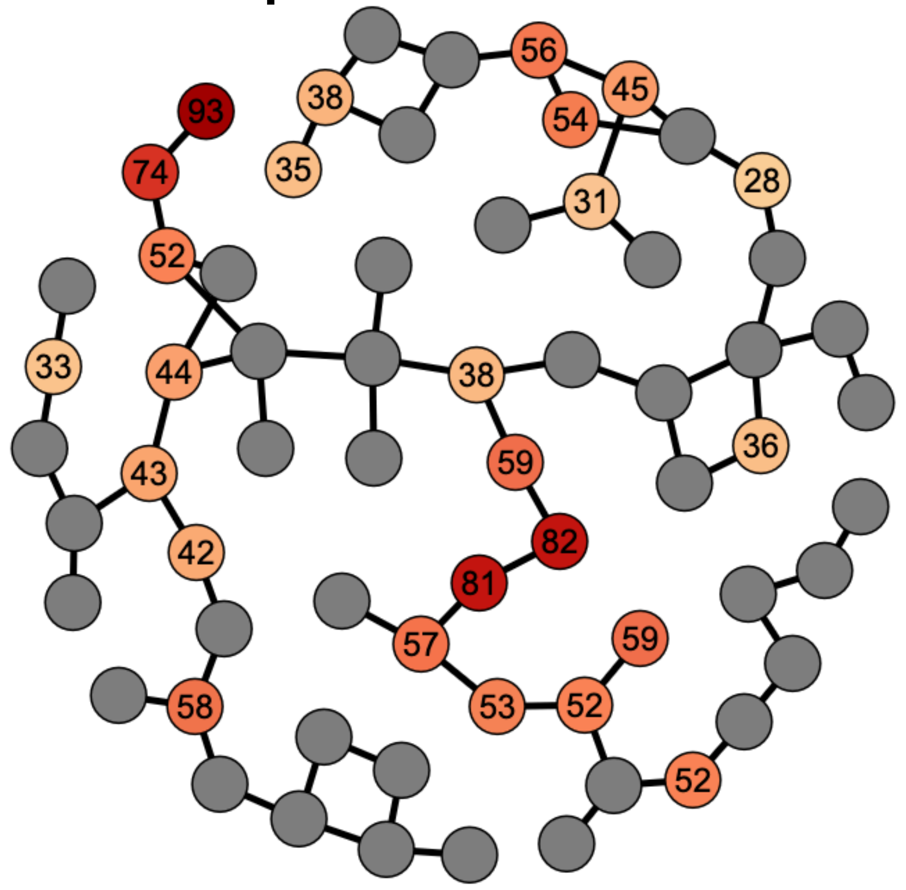
- Generalization guides exploration
  Wu, Schulz, Nelson, Speekenbrink & Meder (*NHB* 2018)
- Developmental trajectory of learning
  Giron*, Ciranka*, Schulz, Van den Bos, Ruggeri, Meder, & Wu (*NHB* 2023)
  Meder, Wu, Schulz & Ruggeri (*DevSci* 2021)
  Schulz, Wu, Ruggeri & Meder (*PsychSci* 2019)
- Search in abstract conceptual spaces
  Wu, Schulz, Garvert, Meder & Schuck (*PLOS Comp Bio, 2020*)
- Graph-structured generalization
  Wu, Schulz & Gershman (*CBB* 2021)

Conceptual features



Wu, Schulz, Garvert, Meder & Schuck (*PLOS Comp Bio 2020*)

Graph structures

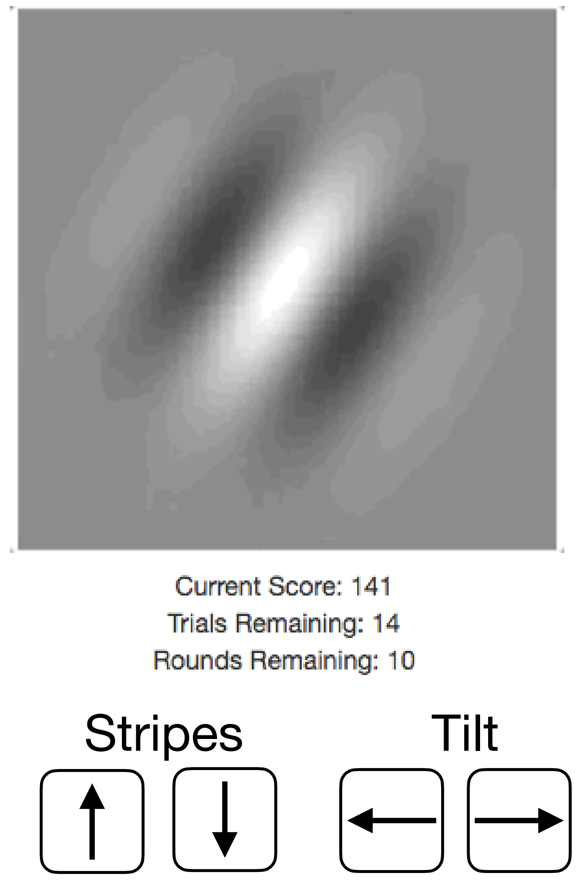


Wu, Schulz & Gershman (*CBB* 2021)

A versatile and robust paradigm

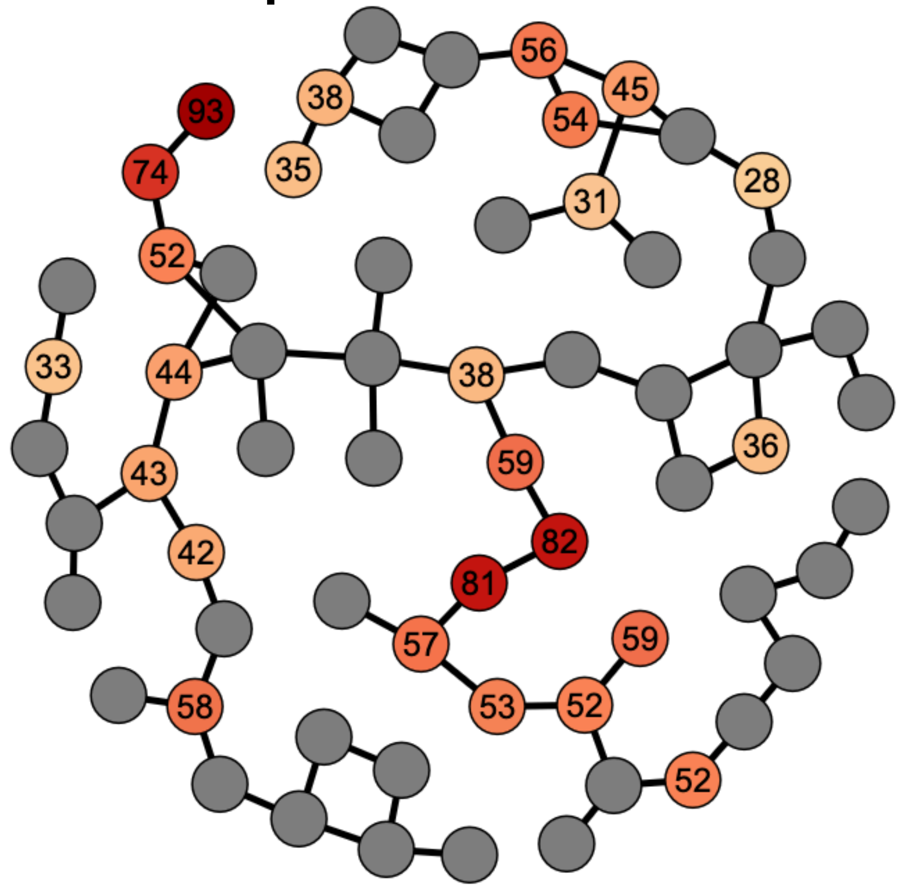
- Generalization guides exploration
  Wu, Schulz, Nelson, Speekenbrink & Meder (*NHB* 2018)
- Developmental trajectory of learning
  Giron*, Ciranka*, Schulz, Van den Bos, Ruggeri, Meder, & Wu (*NHB* 2023)
  Meder, Wu, Schulz & Ruggeri (*DevSci* 2021)
  Schulz, Wu, Ruggeri & Meder (*PsychSci* 2019)
- Search in abstract conceptual spaces
  Wu, Schulz, Garvert, Meder & Schuck (*PLOS Comp Bio, 2020*)
- Graph-structured generalization
  Wu, Schulz & Gershman (*CBB* 2021)
- Safe exploration
  Schulz, Wu, Huys, Krause & Speekenbrink (*Cognitive Science* 2018)
- Forgetful generalization with limited memory
  Breit, Ten, Sakaki, Murayama, & Wu (*KogWiss* 2022)
  Ten, Breit, Sakaki, Murayama & Wu (*in prep*)
- Neural basis for generalization and exploration
  Liebe, Ciranka, Spies, Lanzenburger, & Wu (*in prep*)
  Wong, Moneta, Schuck, Hauser & Wu (*in prep*)
- Social generalization
  Witt, Toyokawa, Lala, Gaissmaier, & Wu (PNAS 2024)
  Wu, Deffner, Kahl, Meder, Ho* & Kurvers* (NatComms *in press*)
  Wu, Ho, Kahl, Leuker, Meder & Kurvers (CogSci 2021)

Conceptual features



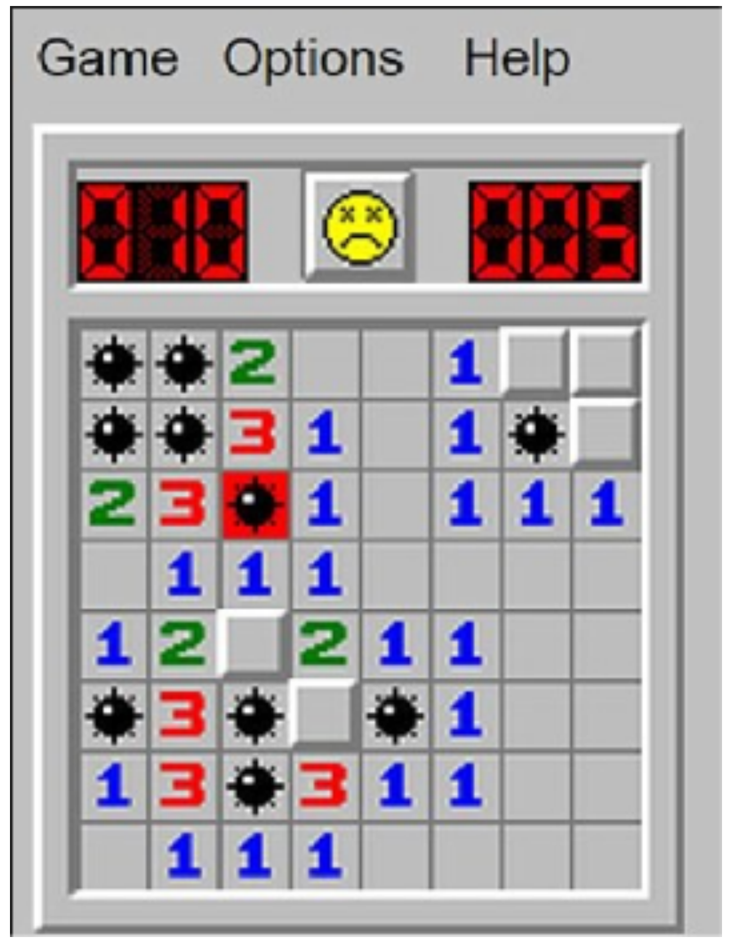
Wu, Schulz, Garvert, Meder & Schuck (*PLOS Comp Bio 2020*)

Graph structures



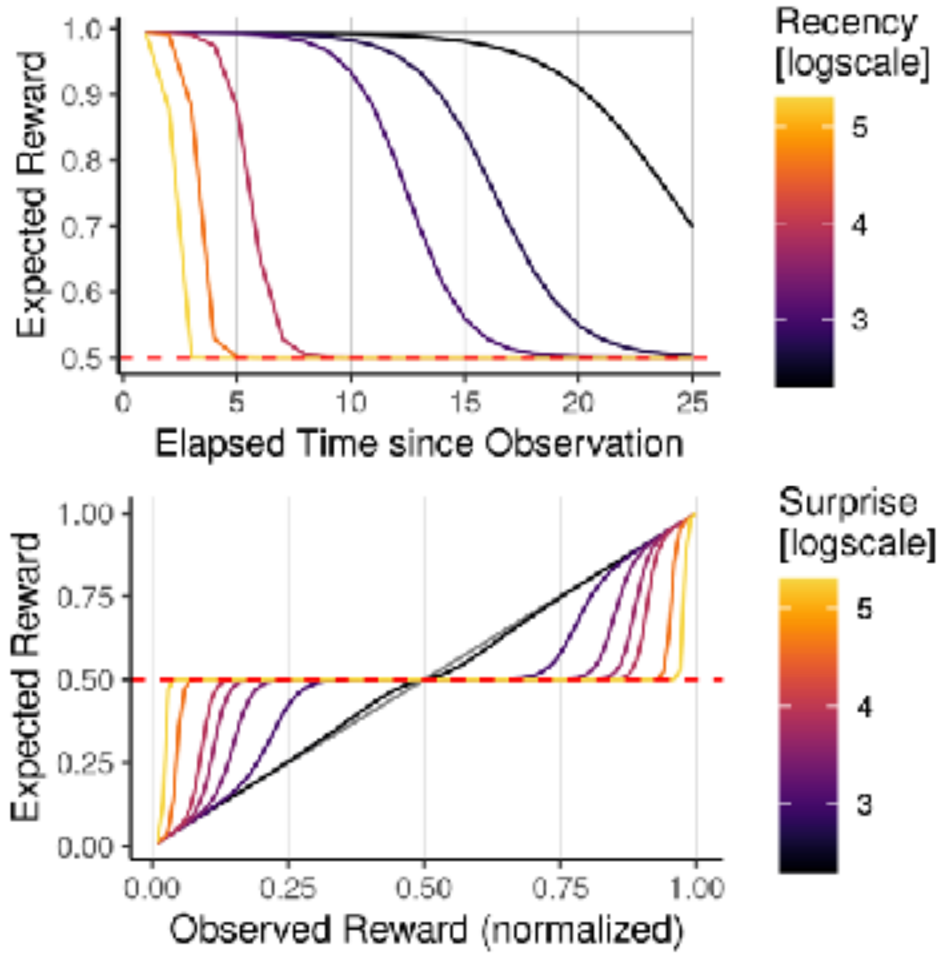
Wu, Schulz & Gershman (*CBB* 2021)

Safe exploration



Schulz, Wu, et al., (*Cognitive Science* 2018)

Forgetful generalization

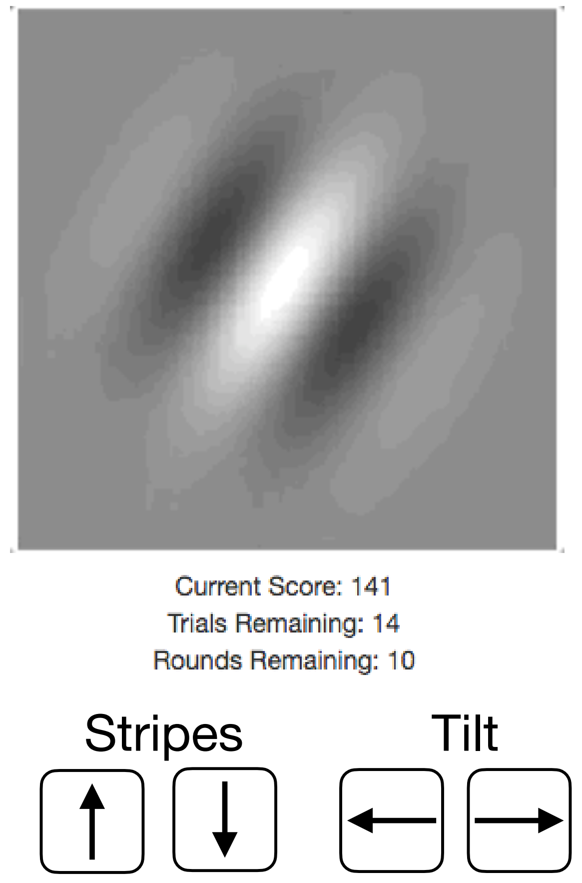


Ten, Breit, Sakaki, Murauama & Wu (*in prep*)

A versatile and robust paradigm

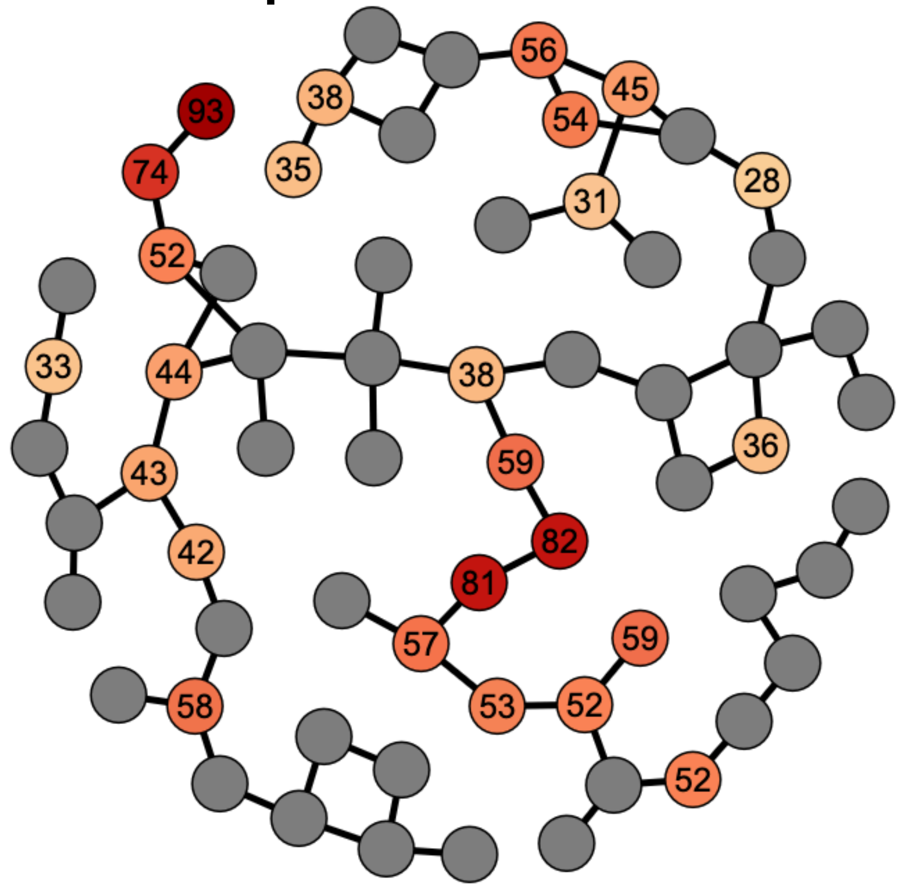
- Generalization guides exploration
  Wu, Schulz, Nelson, Speekenbrink & Meder (*NHB* 2018)
- Developmental trajectory of learning
  Giron*, Ciranka*, Schulz, Van den Bos, Ruggeri, Meder, & Wu (*NHB* 2023)
  Meder, Wu, Schulz & Ruggeri (*DevSci* 2021)
  Schulz, Wu, Ruggeri & Meder (*PsychSci* 2019)
- Search in abstract conceptual spaces
  Wu, Schulz, Garvert, Meder & Schuck (*PLOS Comp Bio, 2020*)
- **Graph-structured generalization**
  Wu, Schulz & Gershman (*CBB* 2021)
- Safe exploration
  Schulz, Wu, Huys, Krause & Speekenbrink (*Cognitive Science* 2018)
- Forgetful generalization with limited memory
  Breit, Ten, Sakaki, Murayama, & Wu (*KogWiss* 2022)
  Ten, Breit, Sakaki, Murayama & Wu (*in prep*)
- Neural basis for generalization and exploration
  Liebe, Ciranka, Spies, Lanzenburger, & Wu (*in prep*)
  Wong, Moneta, Schuck, Hauser & Wu (*in prep*)
- Social generalization
  Witt, Toyokawa, Lala, Gaissmaier, & Wu (PNAS 2024)
  Wu, Deffner, Kahl, Meder, Ho* & Kurvers* (NatComms *in press*)
  Wu, Ho, Kahl, Leuker, Meder & Kurvers (CogSci 2021)

Conceptual features



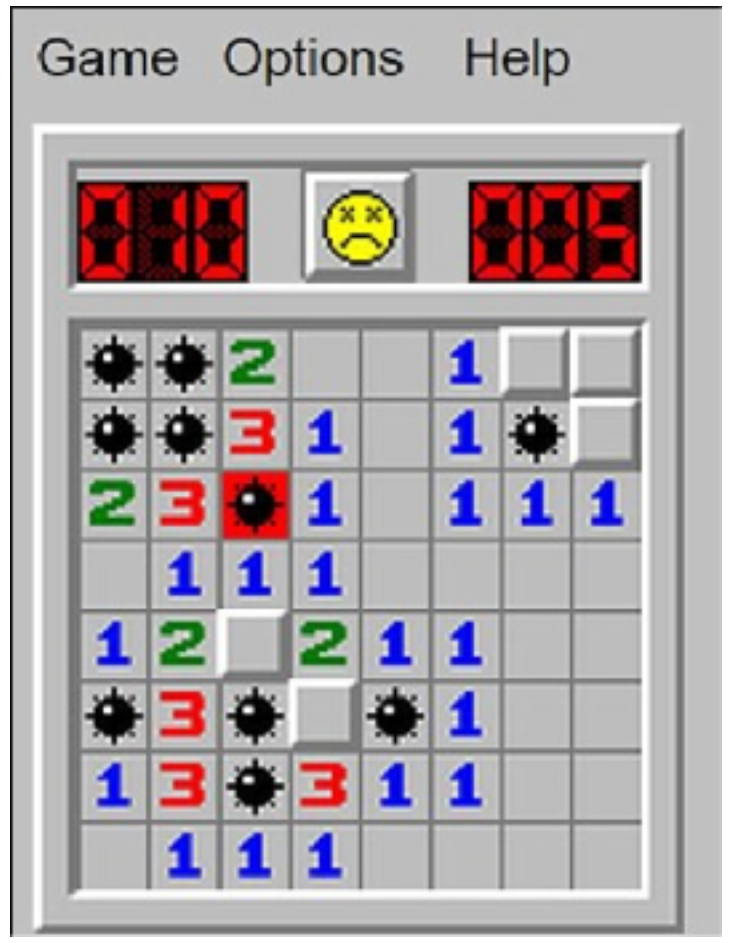
Wu, Schulz, Garvert, Meder & Schuck (*PLOS Comp Bio 2020*)

Graph structures



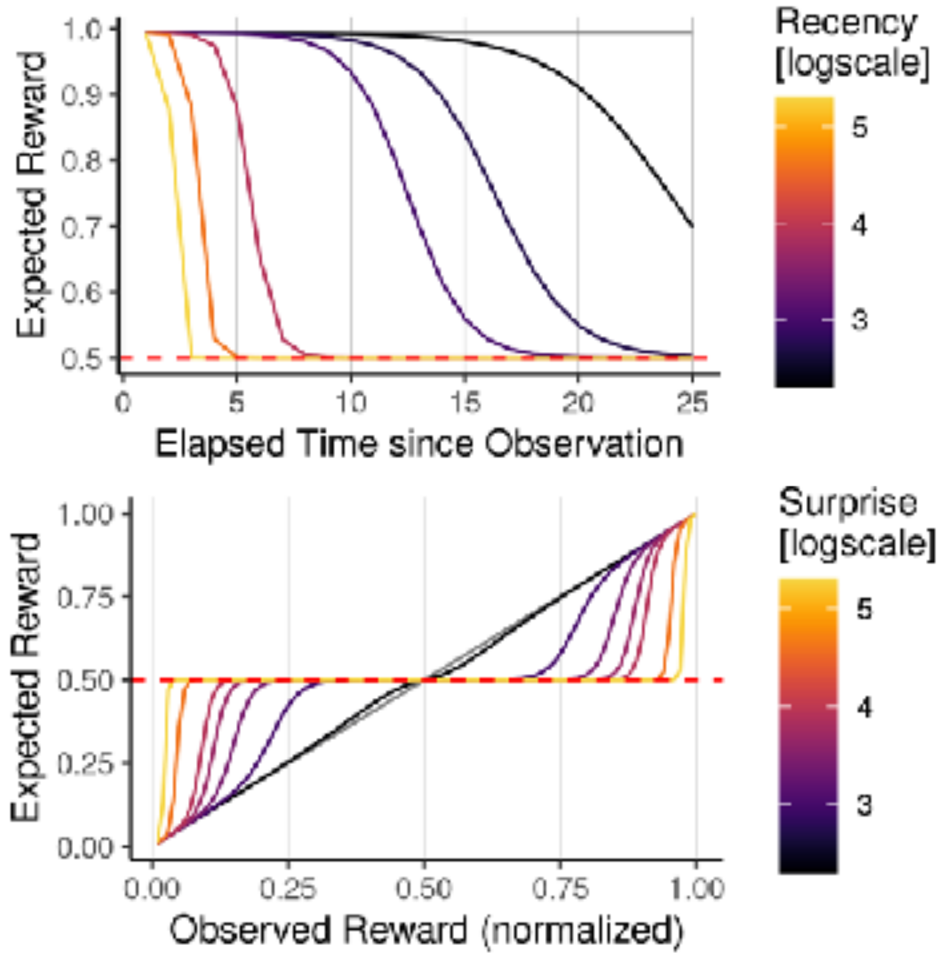
Wu, Schulz & Gershman (*CBB* 2021)

Safe exploration

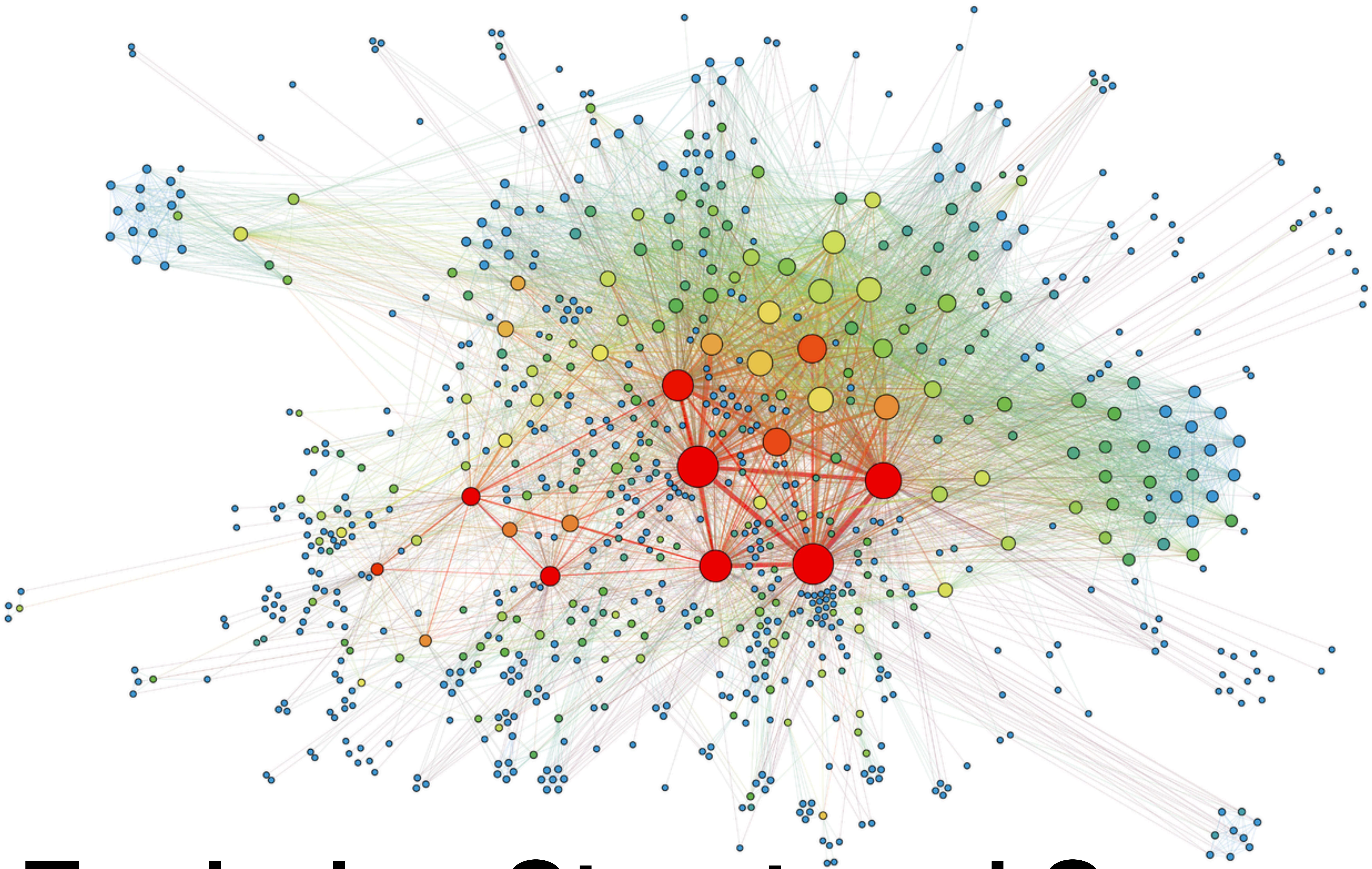


Schulz, Wu, et al., (*Cognitive Science* 2018)

Forgetful generalization

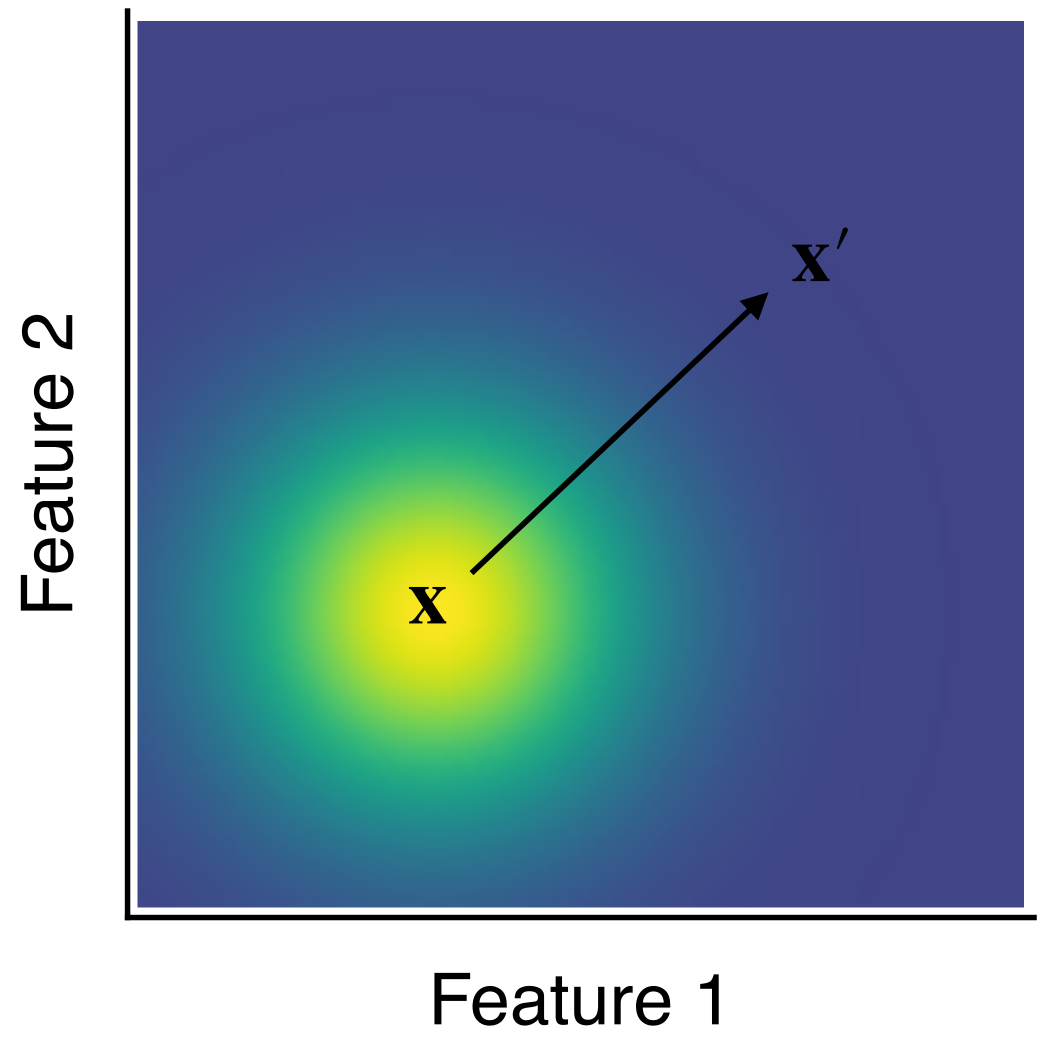


Ten, Breit, Sakaki, Murauama & Wu (*in prep*)

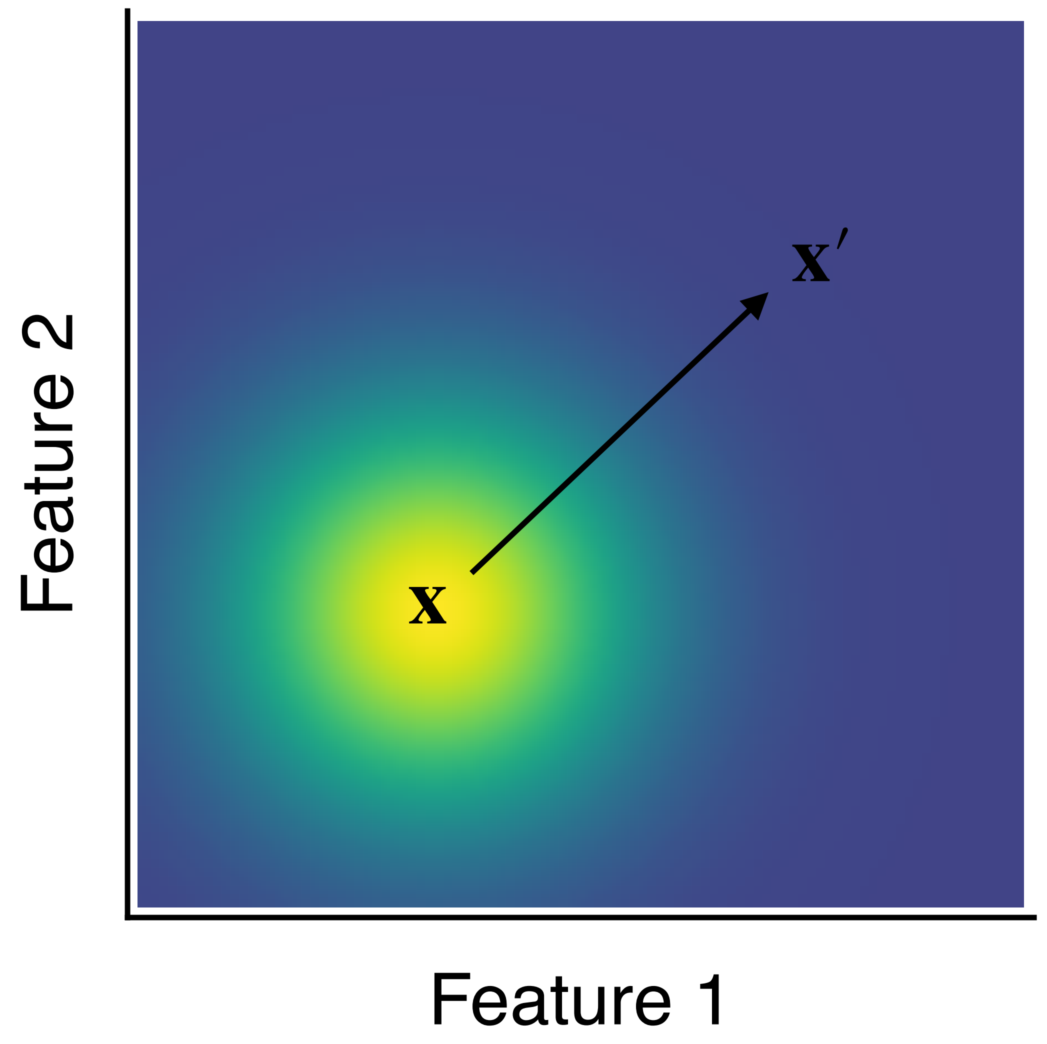


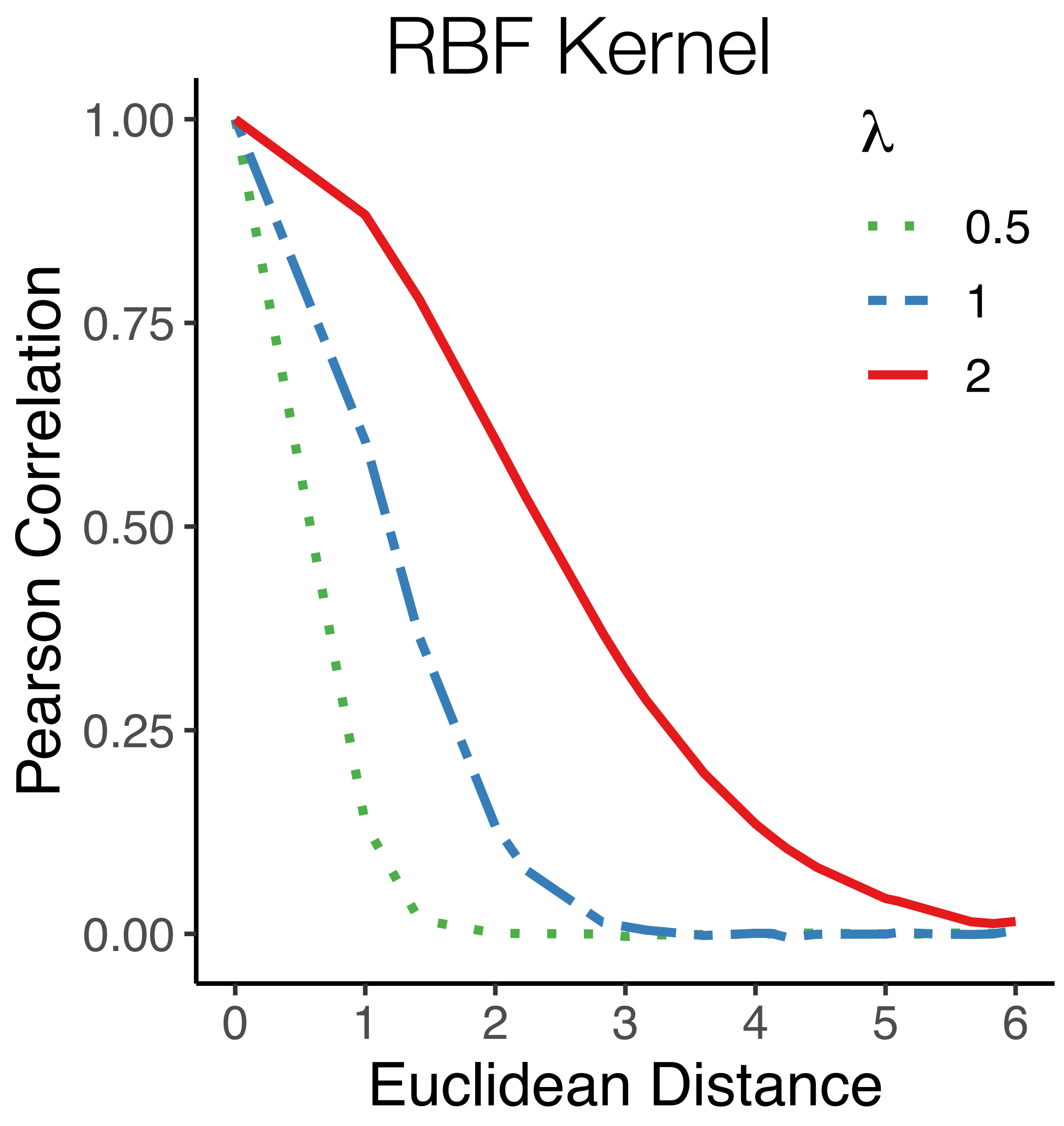
Exploring Structured Spaces

From continuous to structured spaces

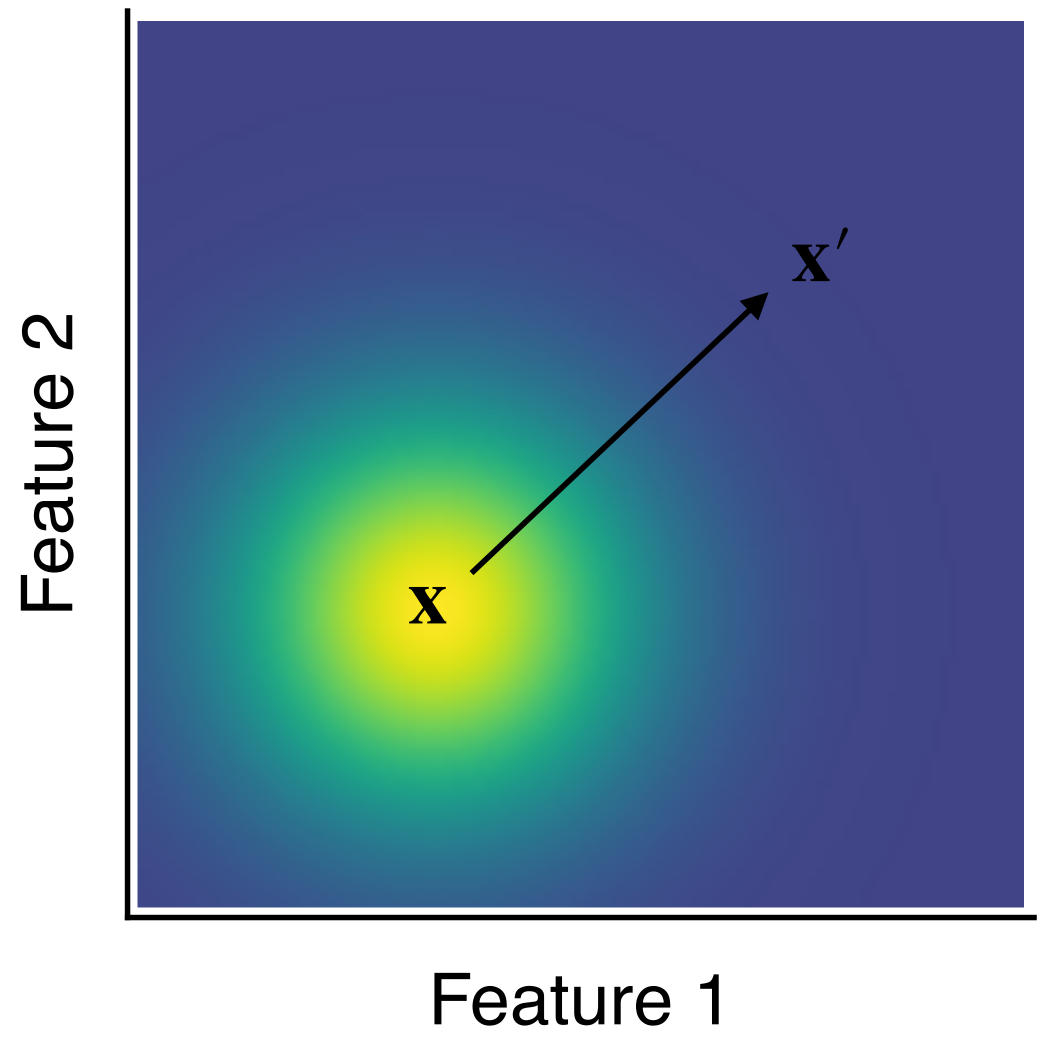


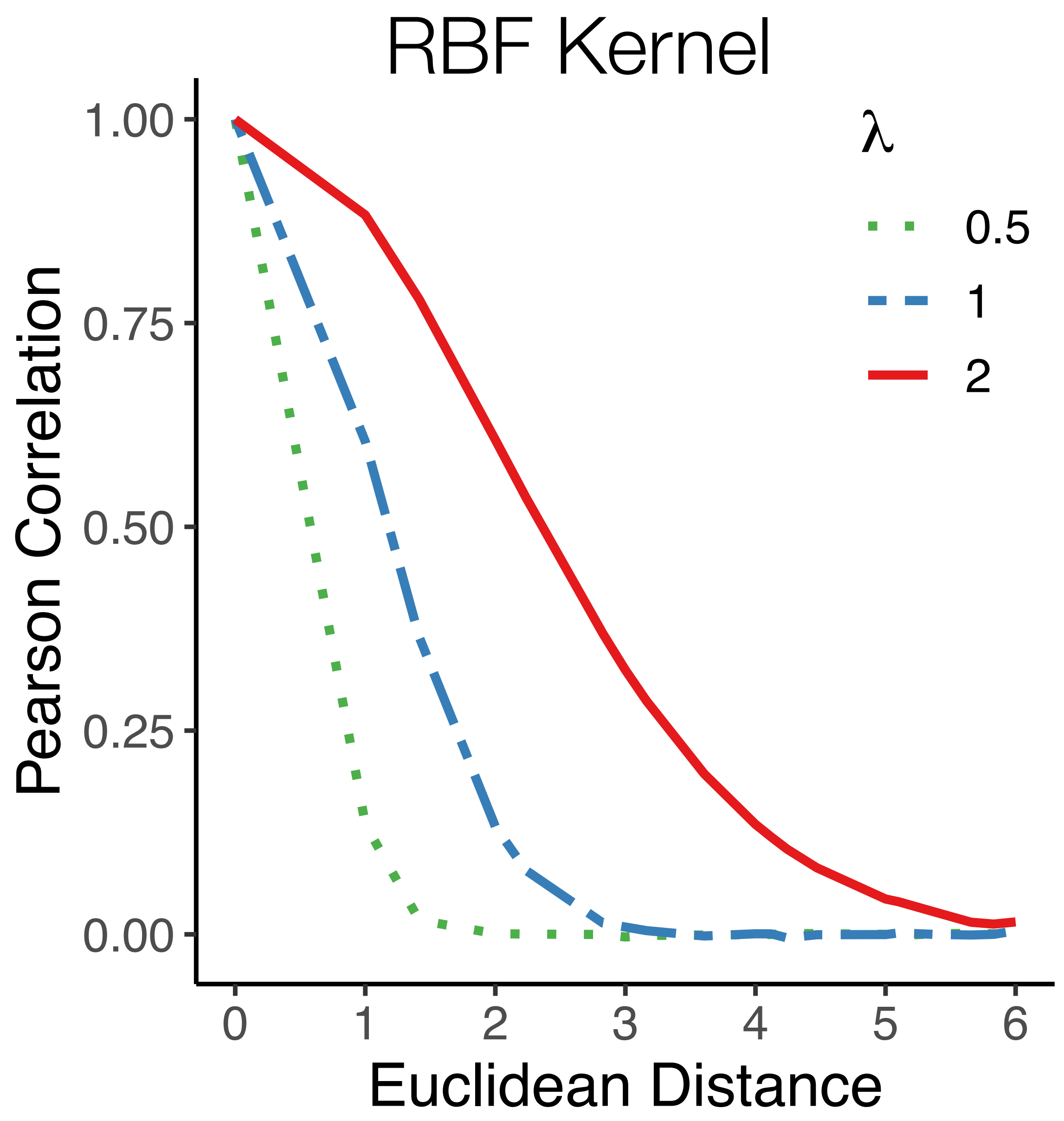
From continuous to structured spaces

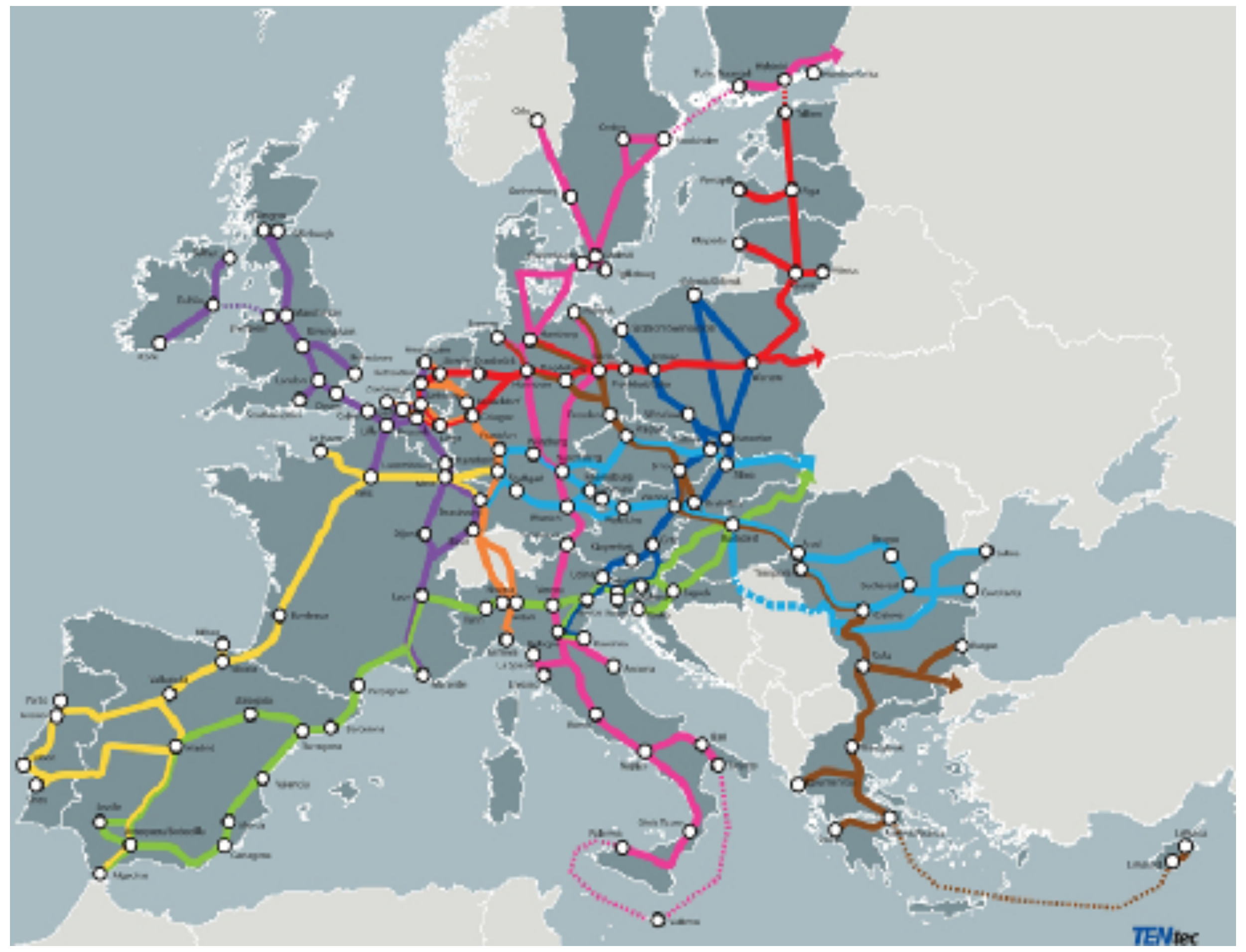




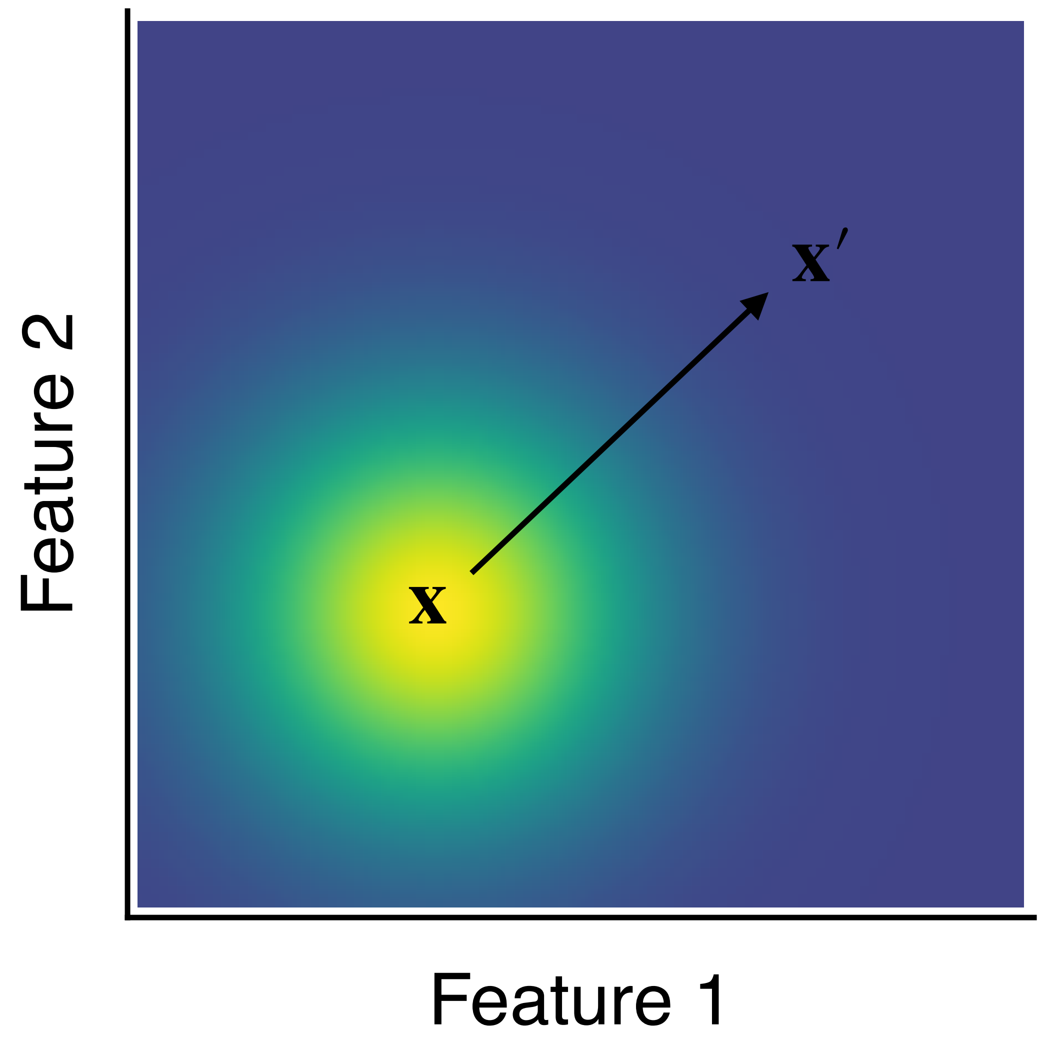
From continuous to structured spaces

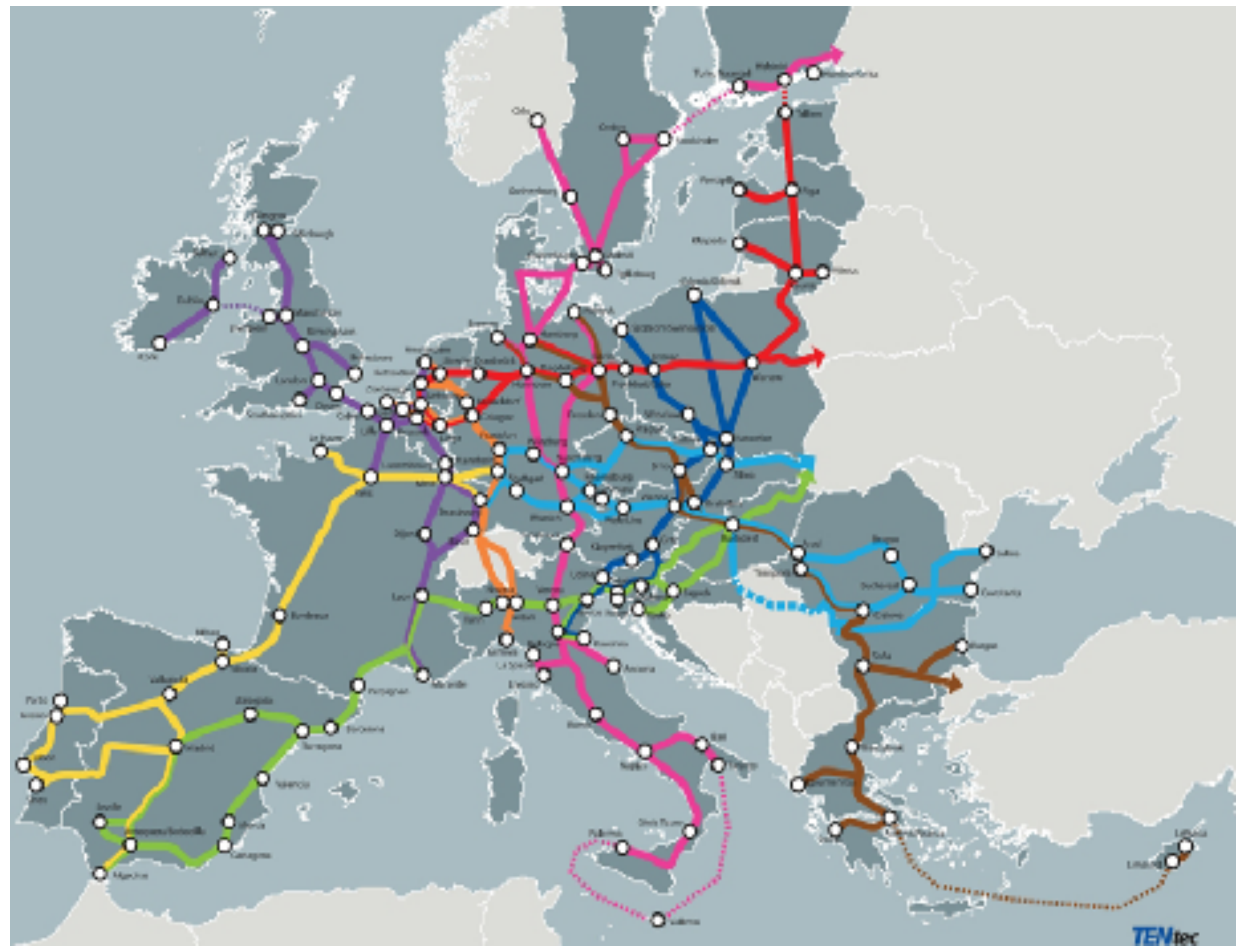


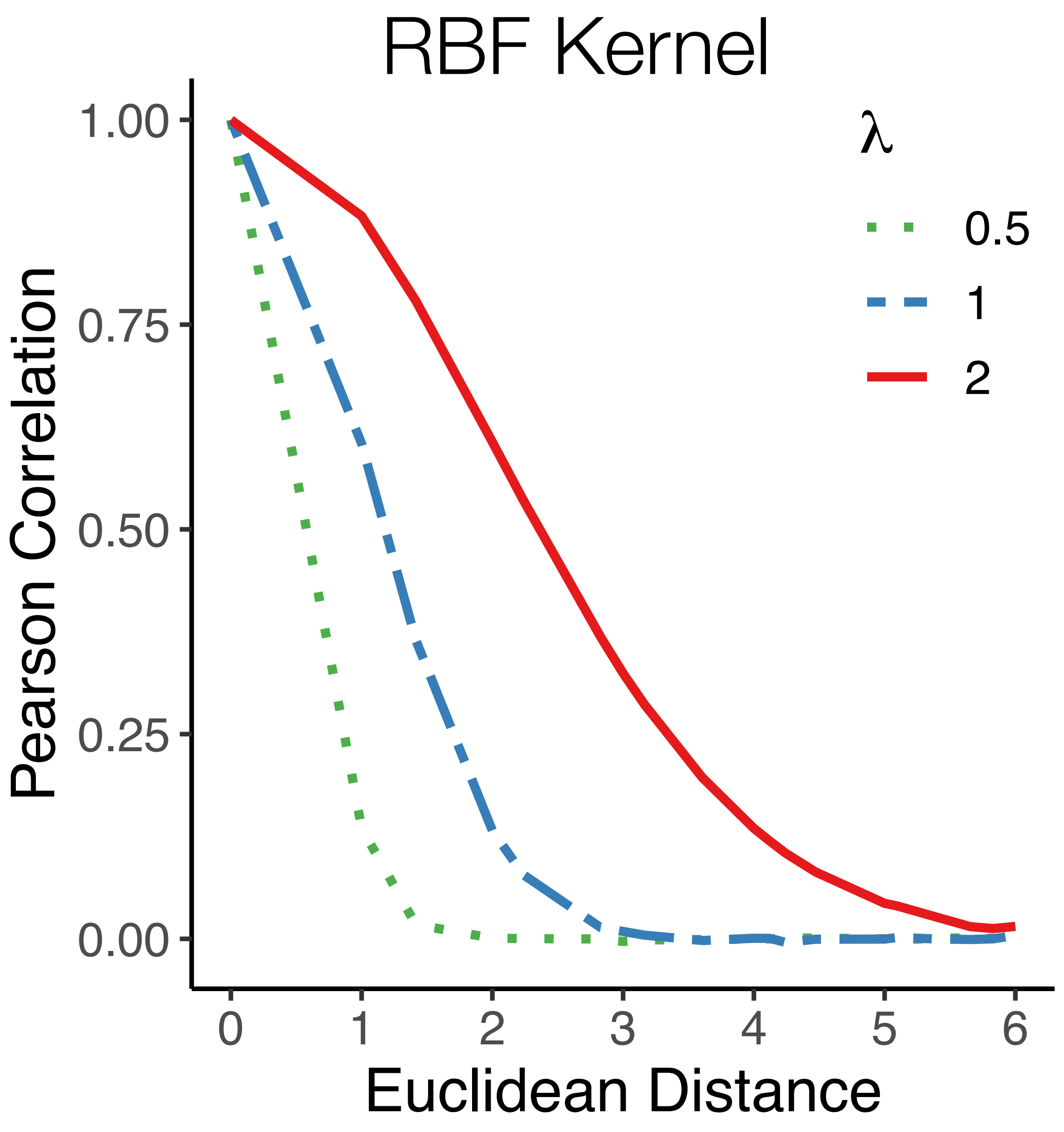


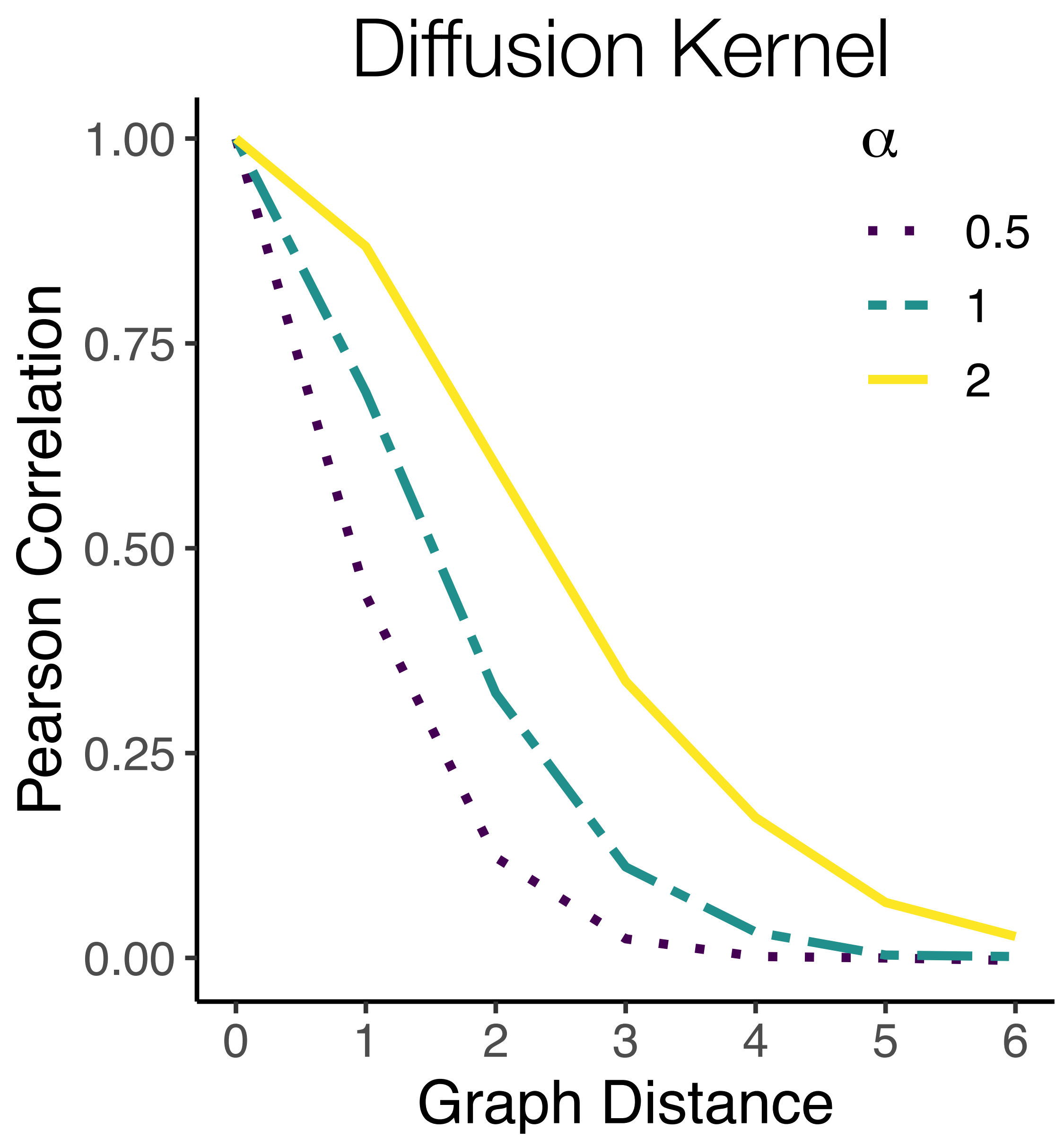


From continuous to structured spaces

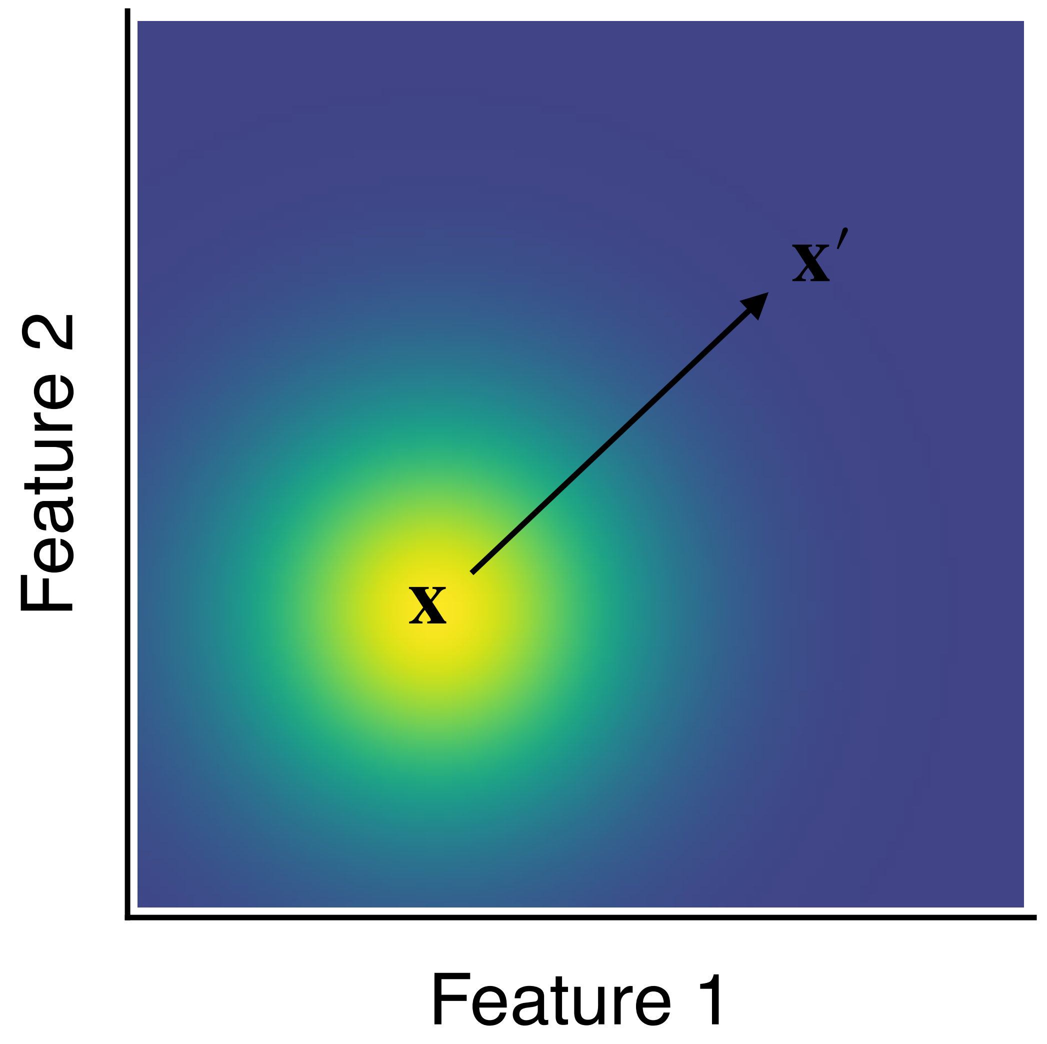


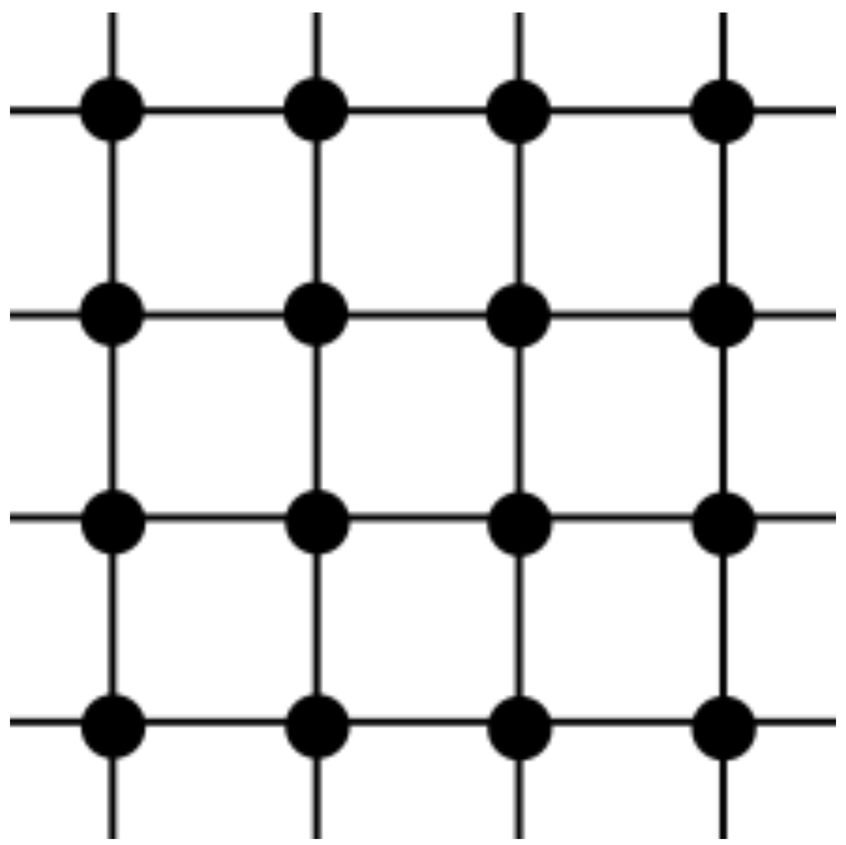


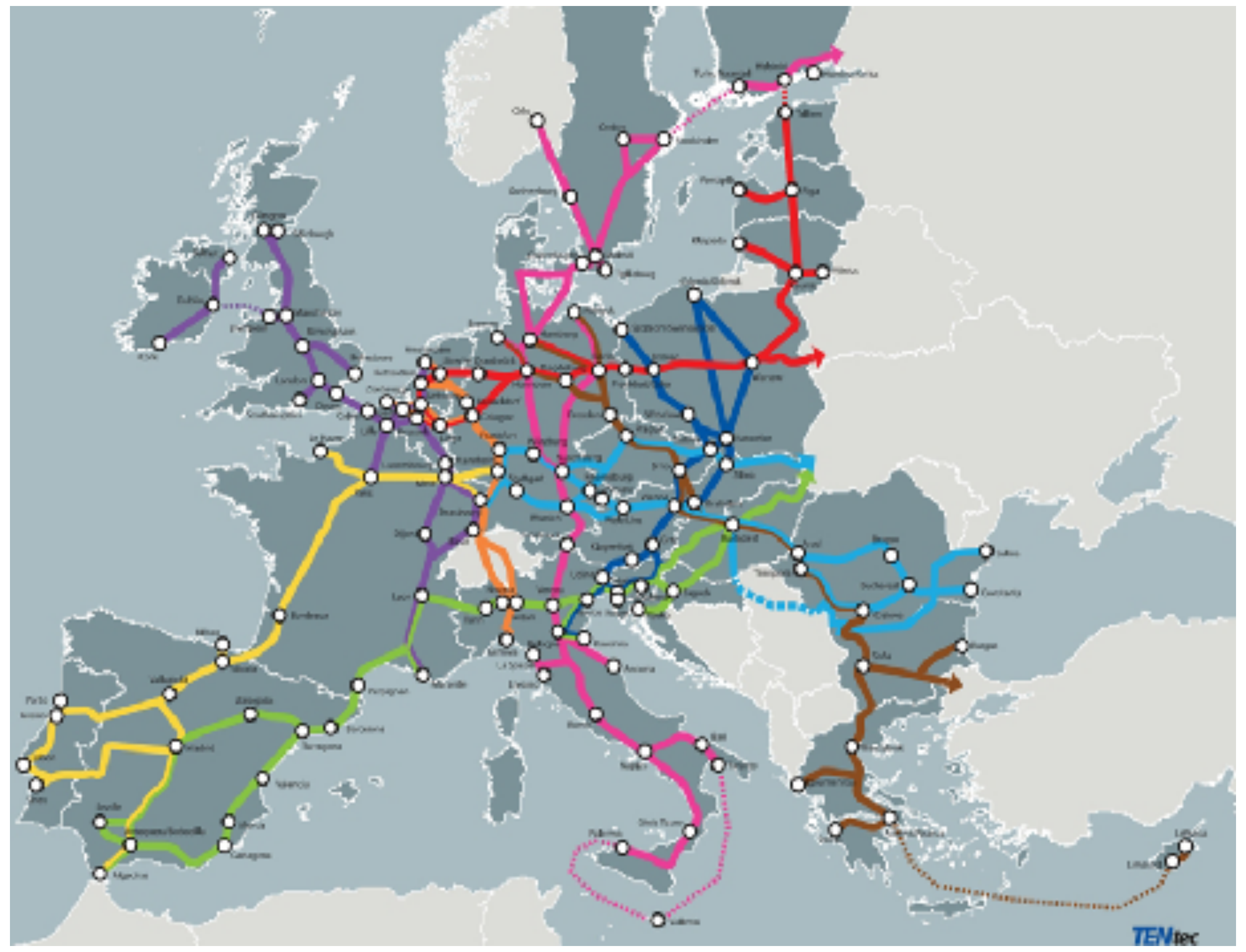


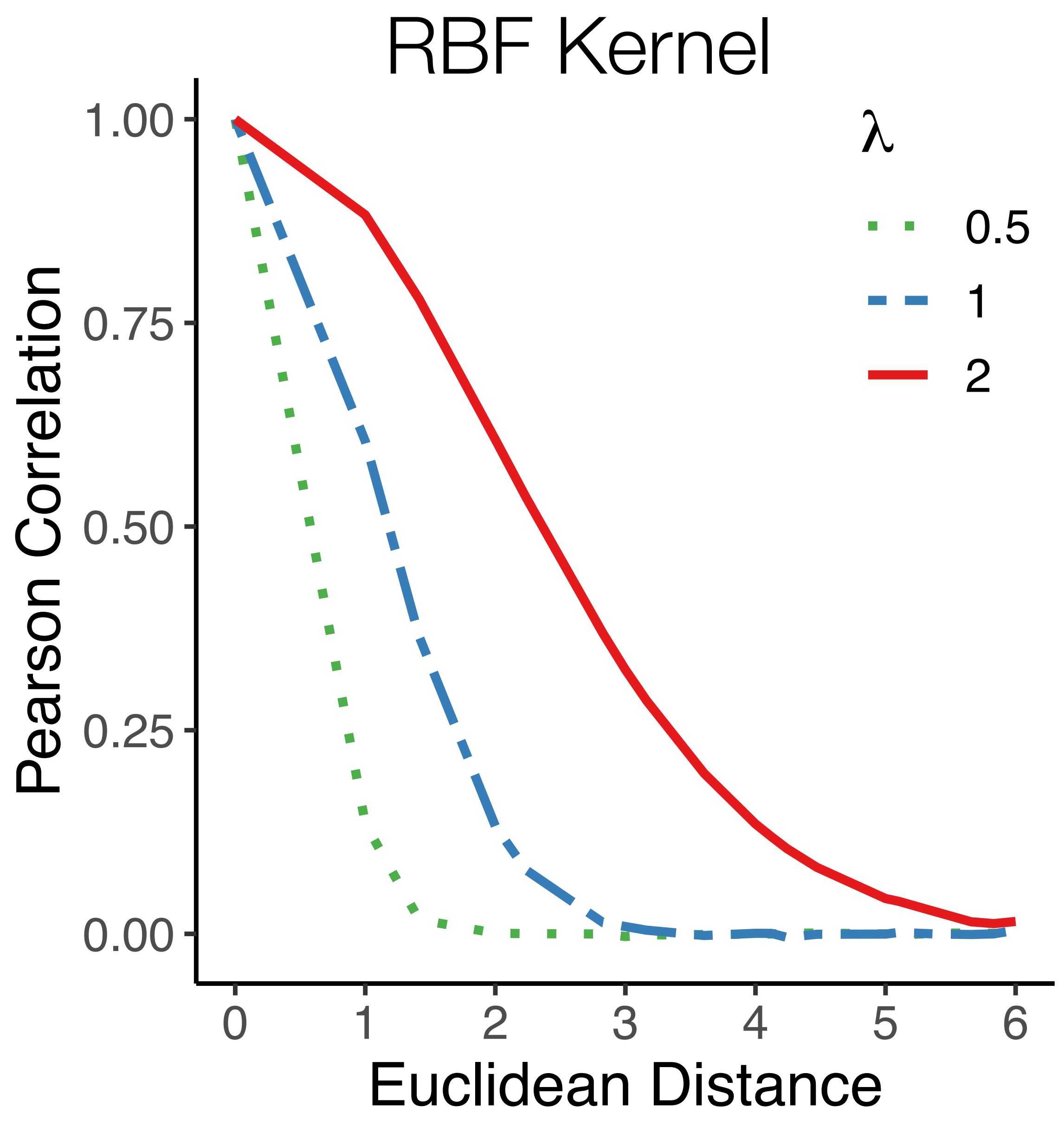


From continuous to structured spaces

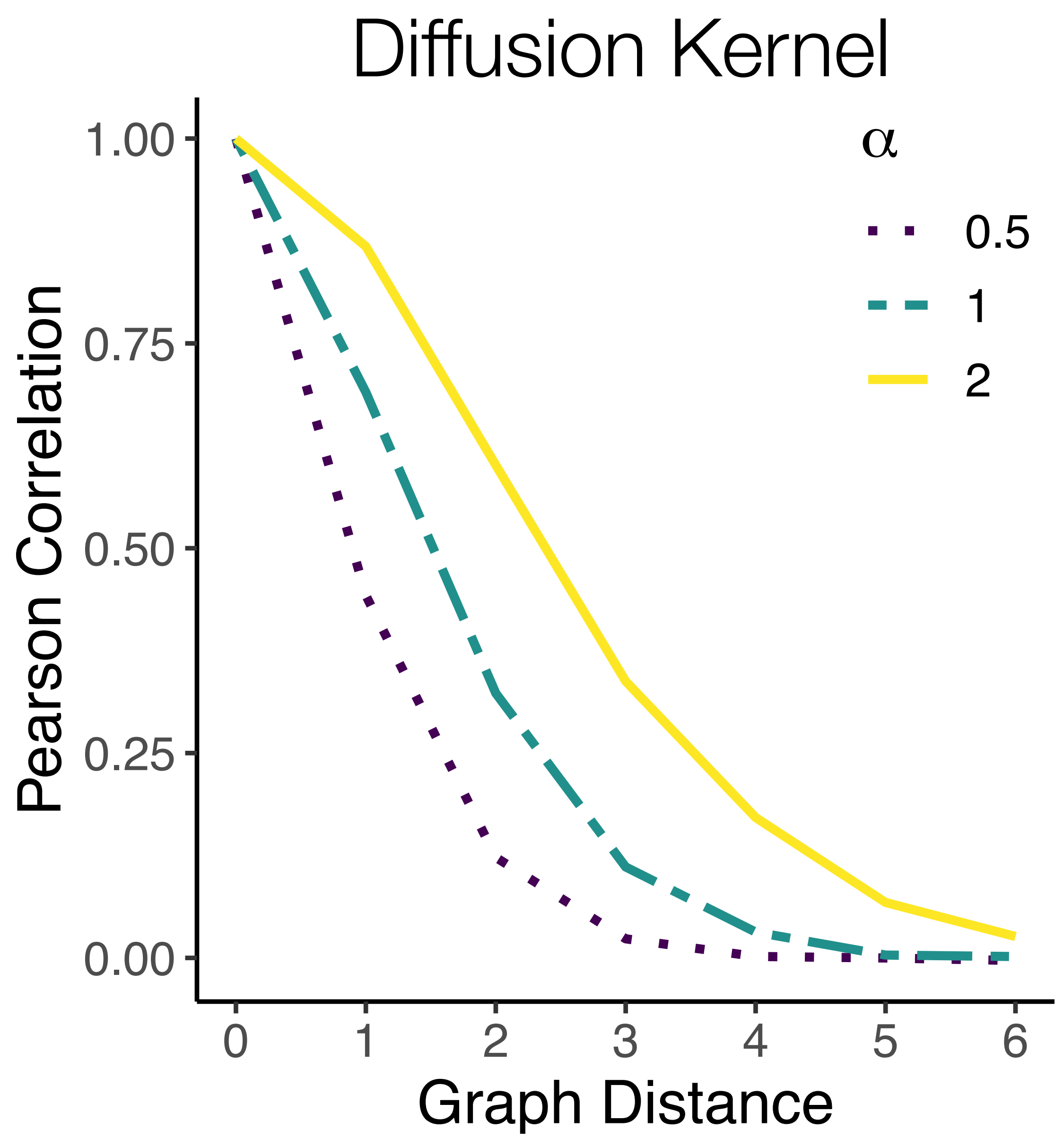






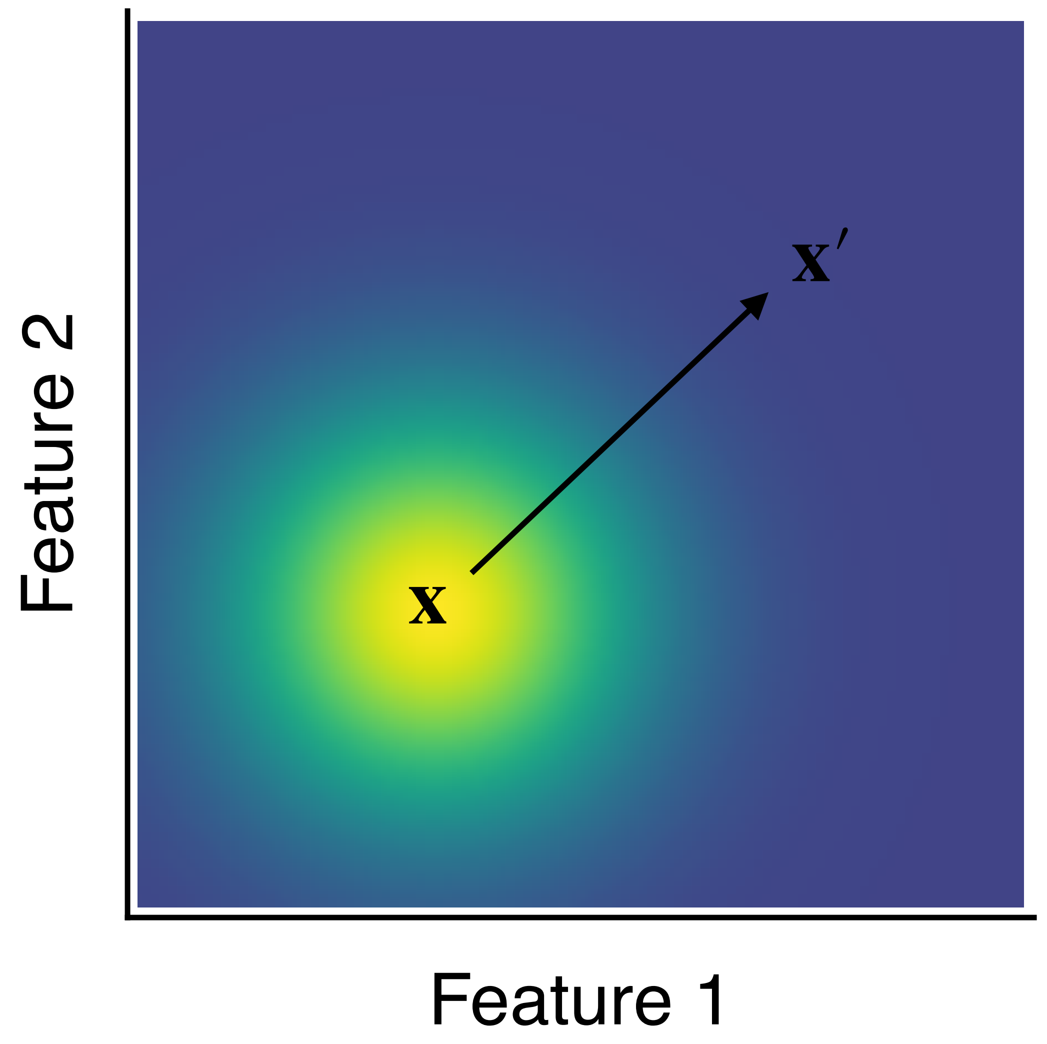


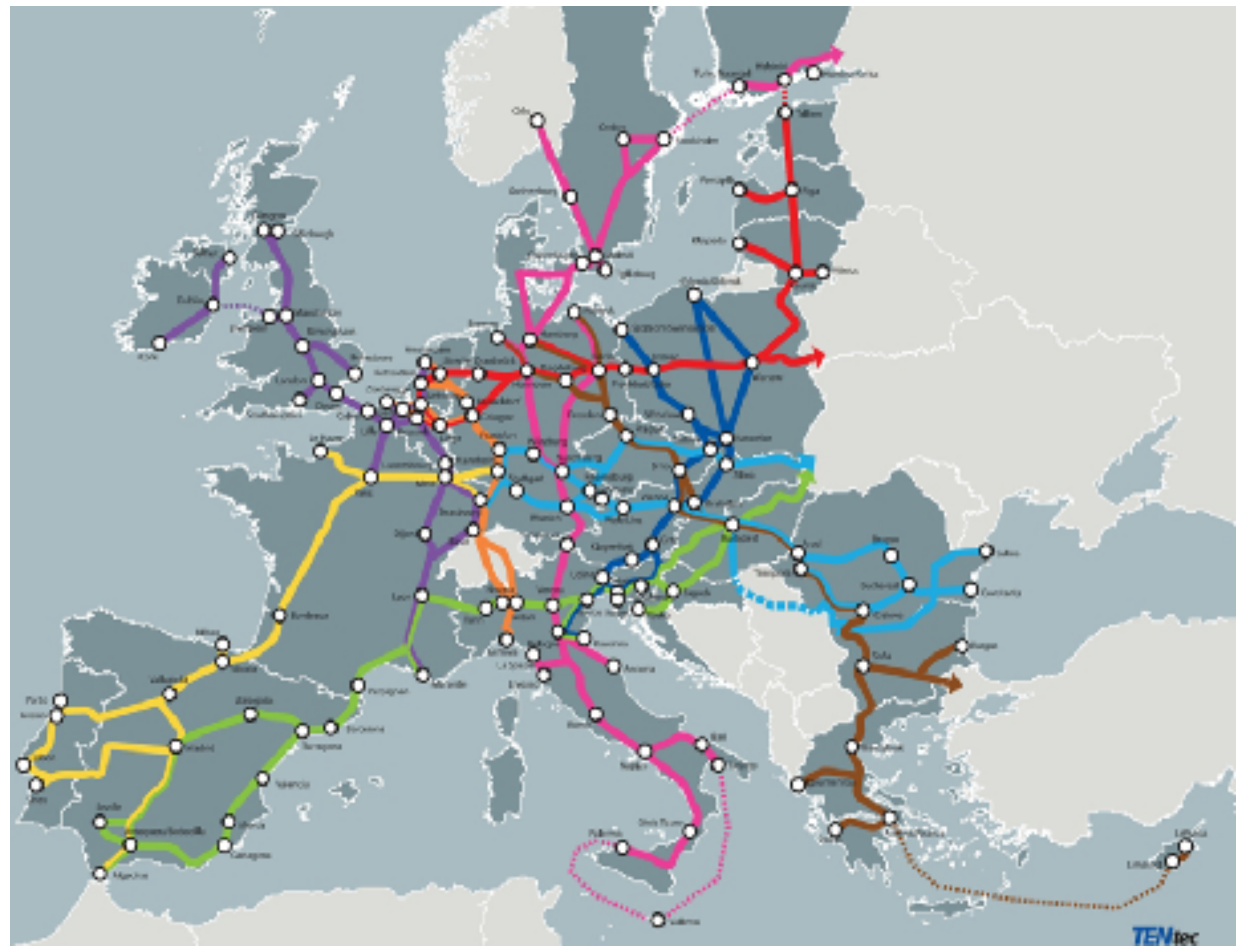
=

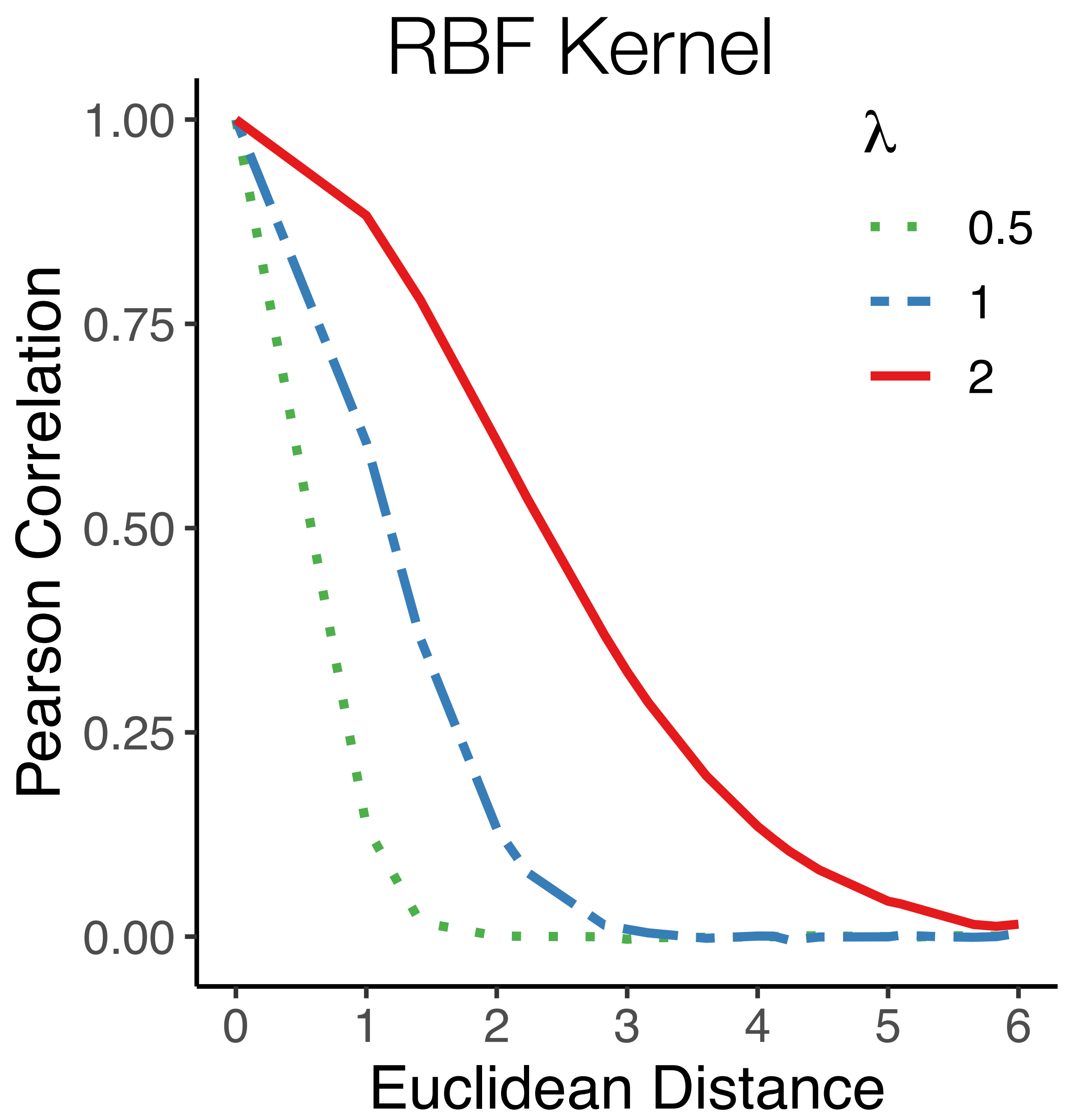


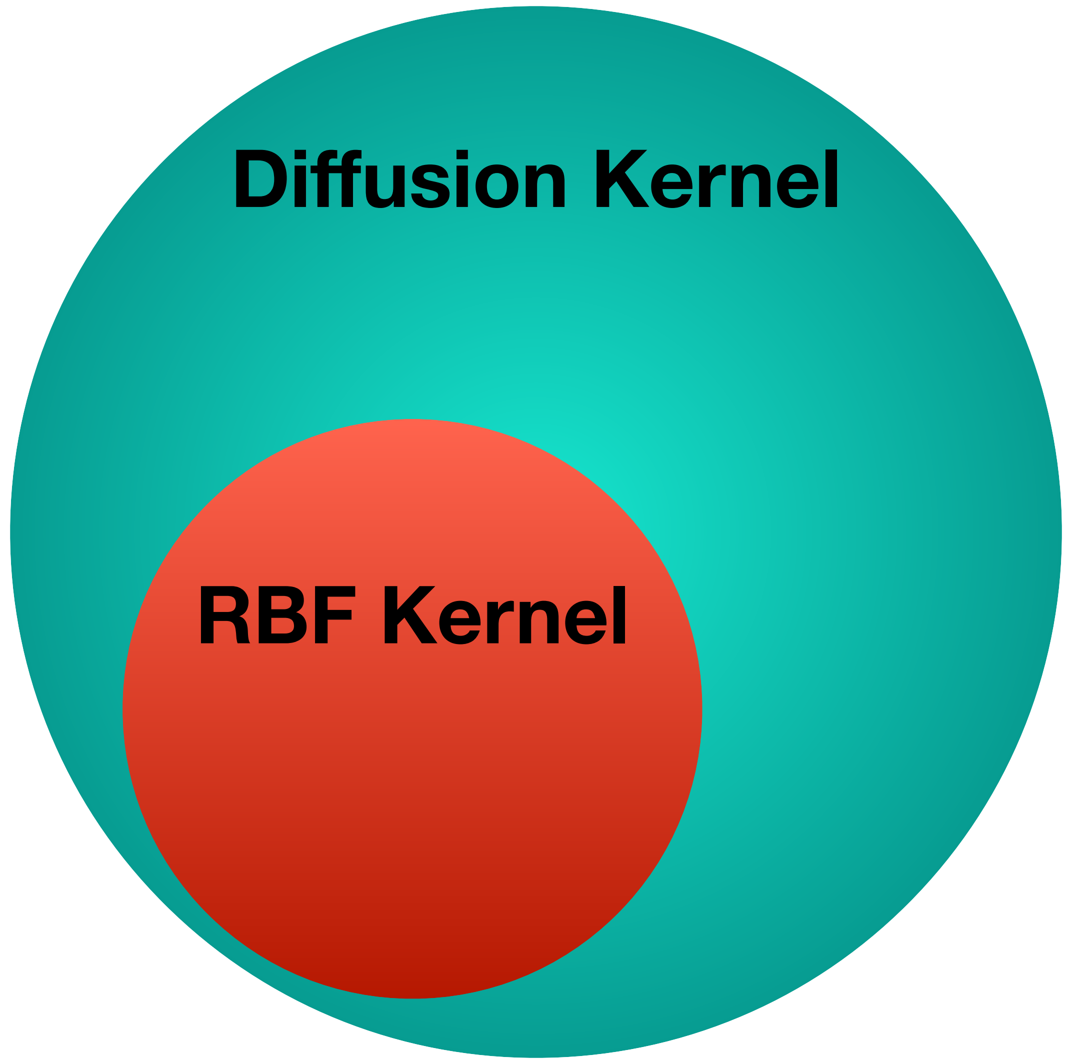
Wu, Schulz, Gershman (*CBB* 2020)

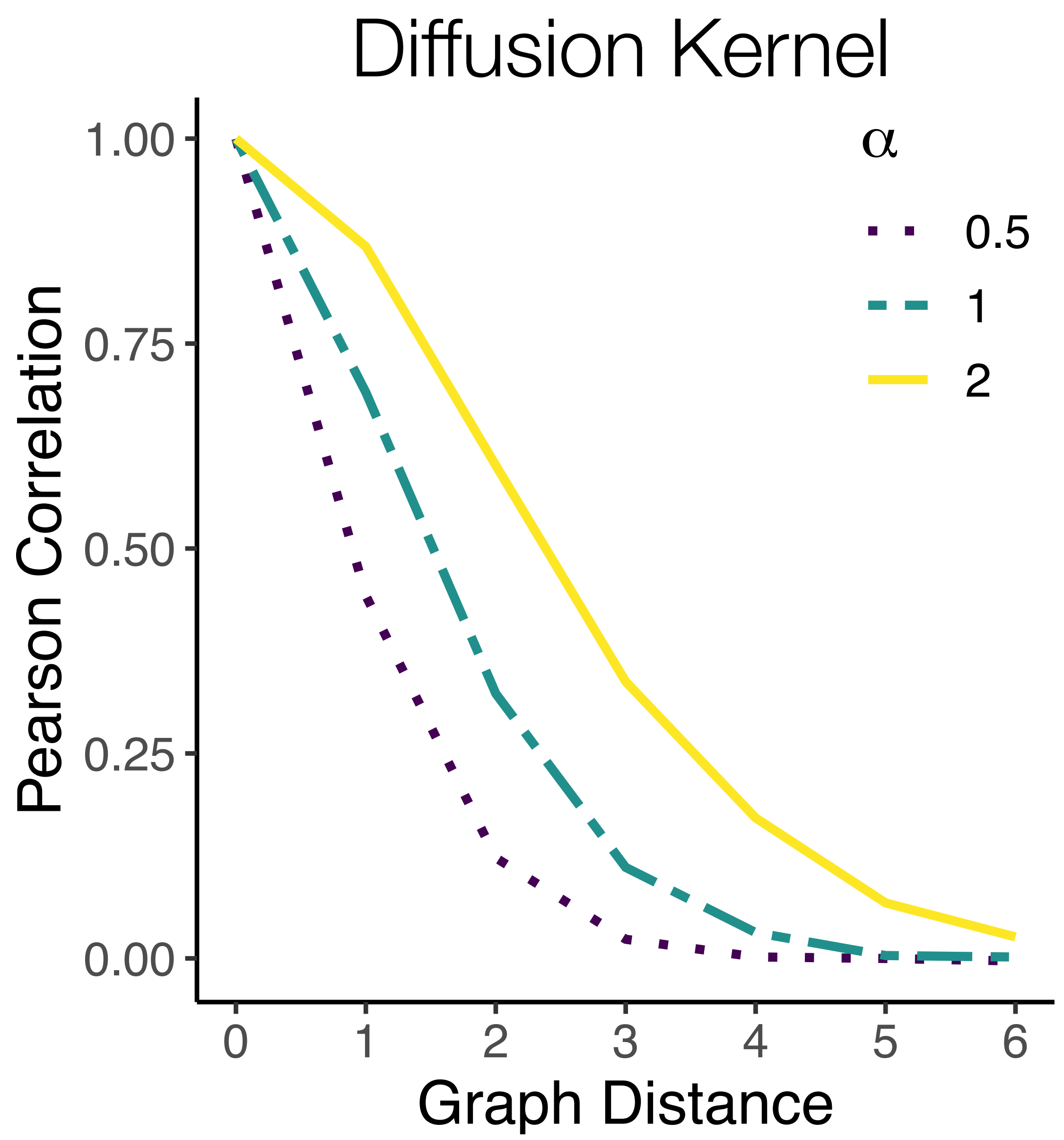
From continuous to structured spaces





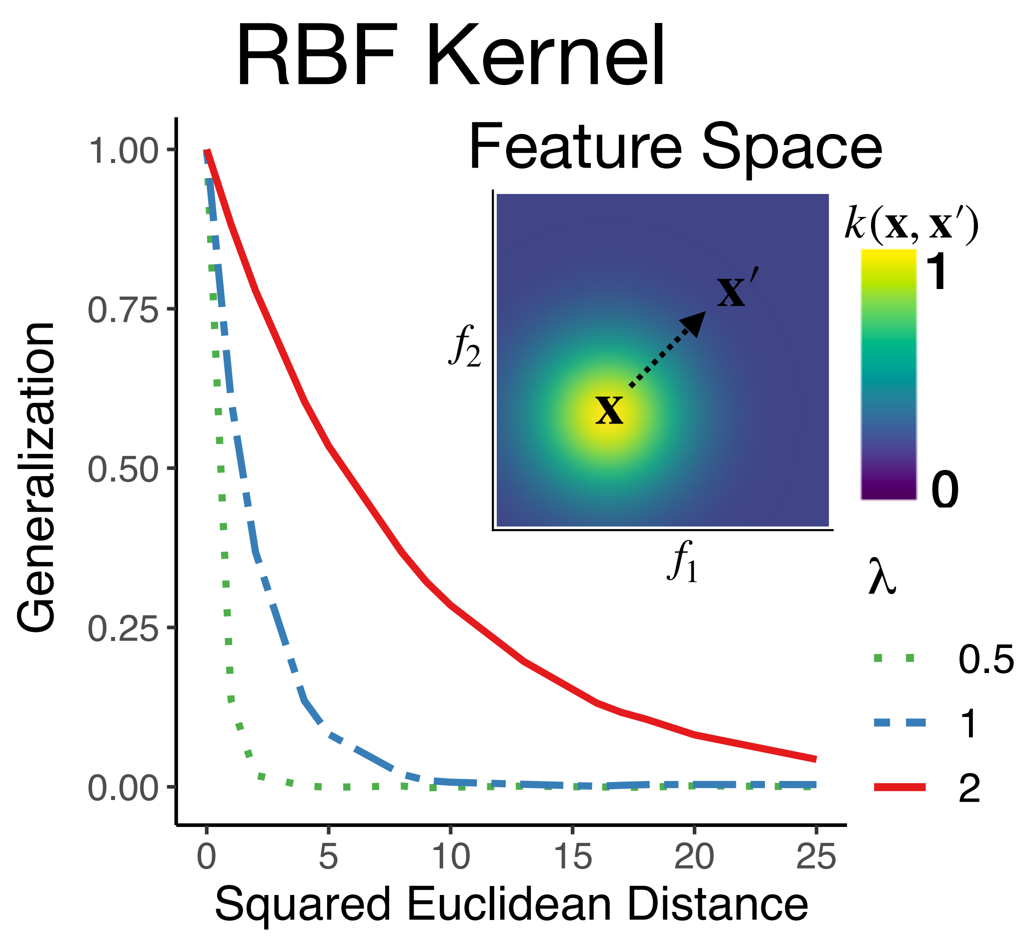






Similarity can also capture relational structure

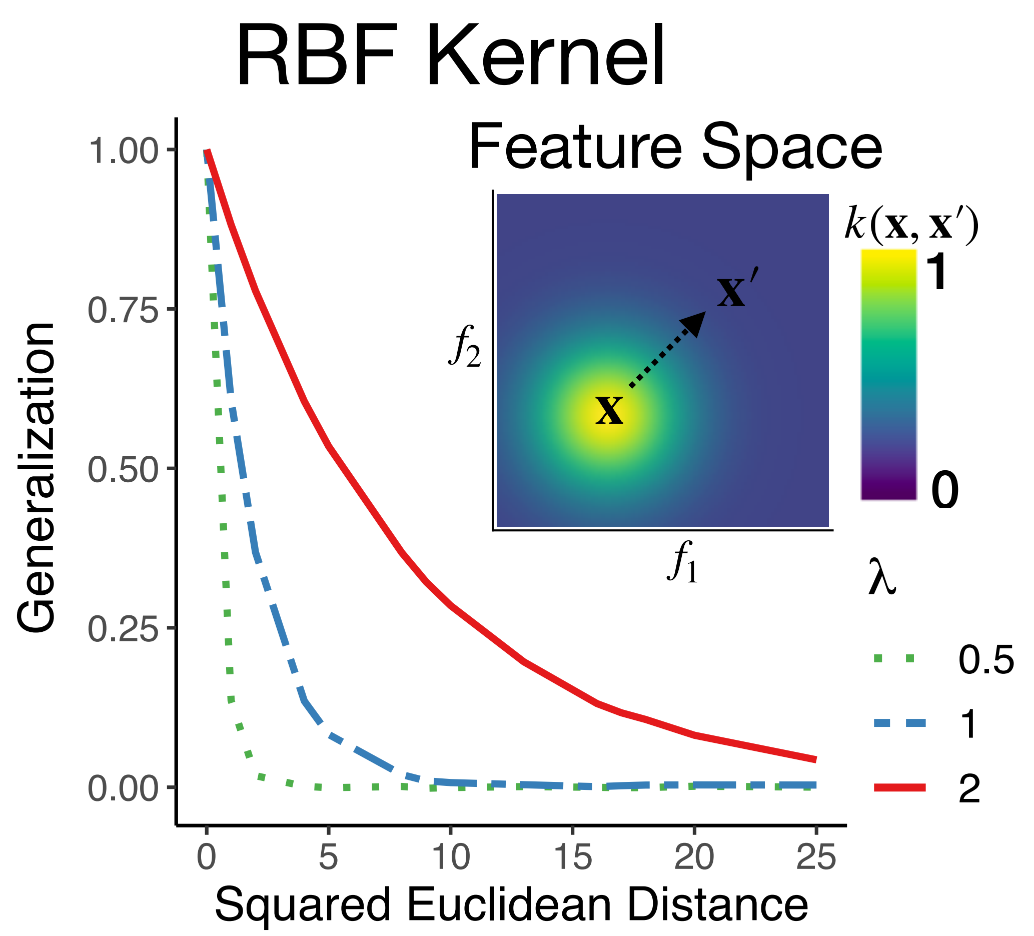
- The RBF kernel, like most classic accounts, represent similarity as distance in feature space
  - Learns smooth functions in a continuous domain

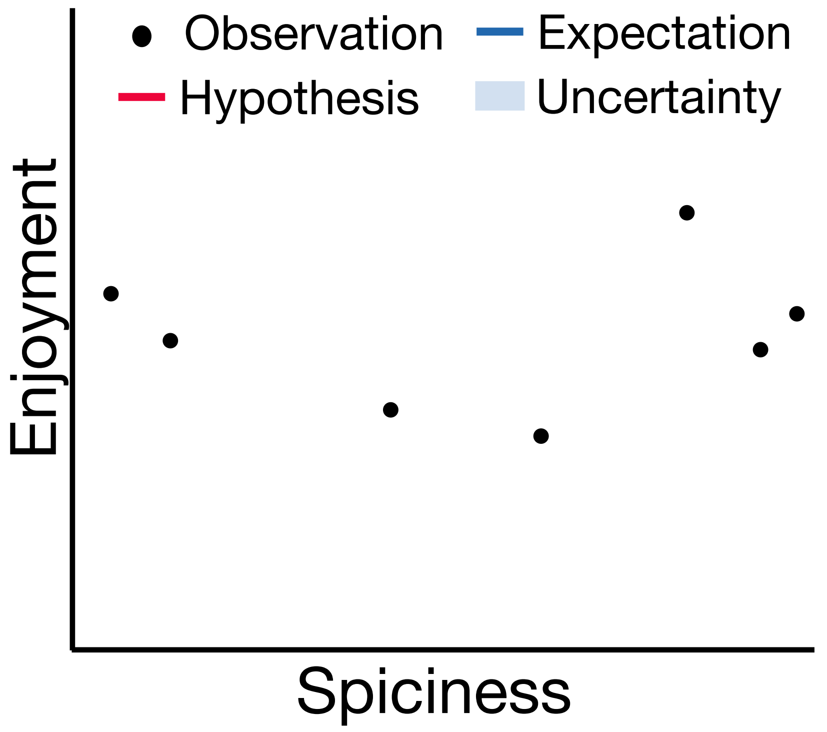


Wu et al., (*CBB* 2021)

Similarity can also capture relational structure

- The RBF kernel, like most classic accounts, represent similarity as distance in feature space
  - Learns smooth functions in a continuous domain

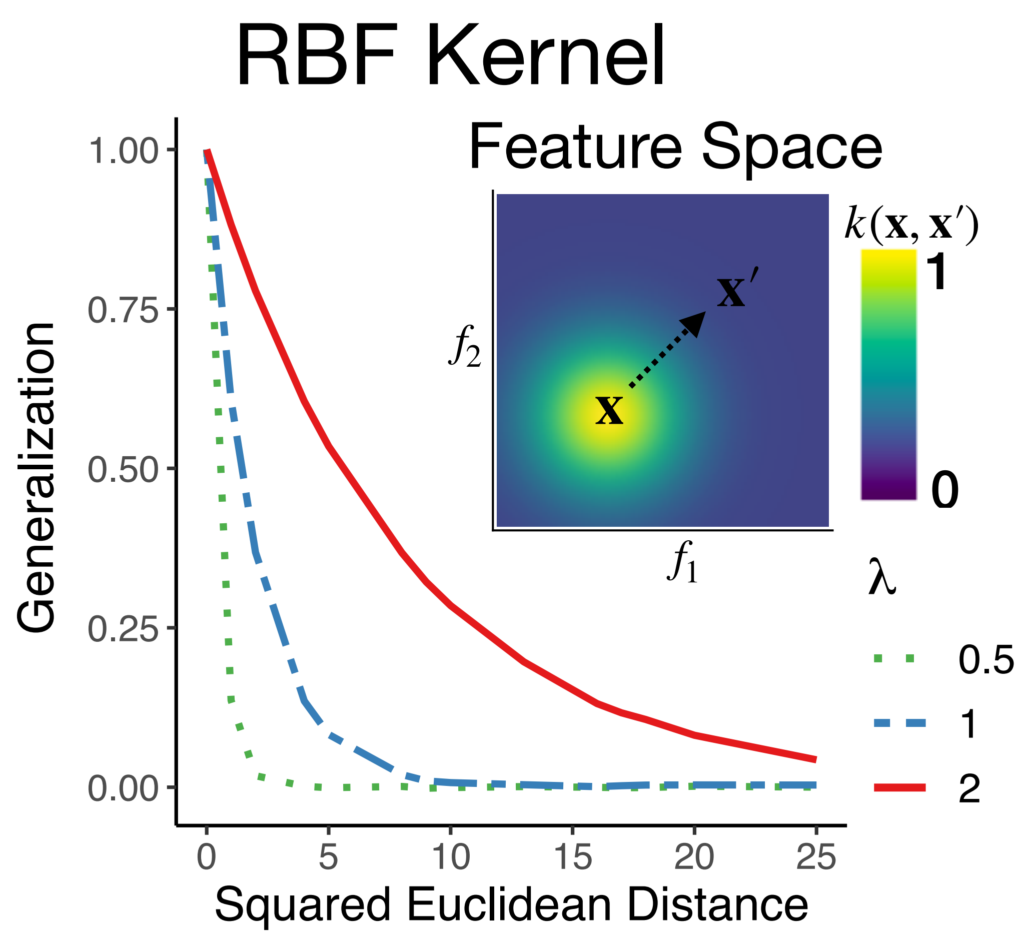


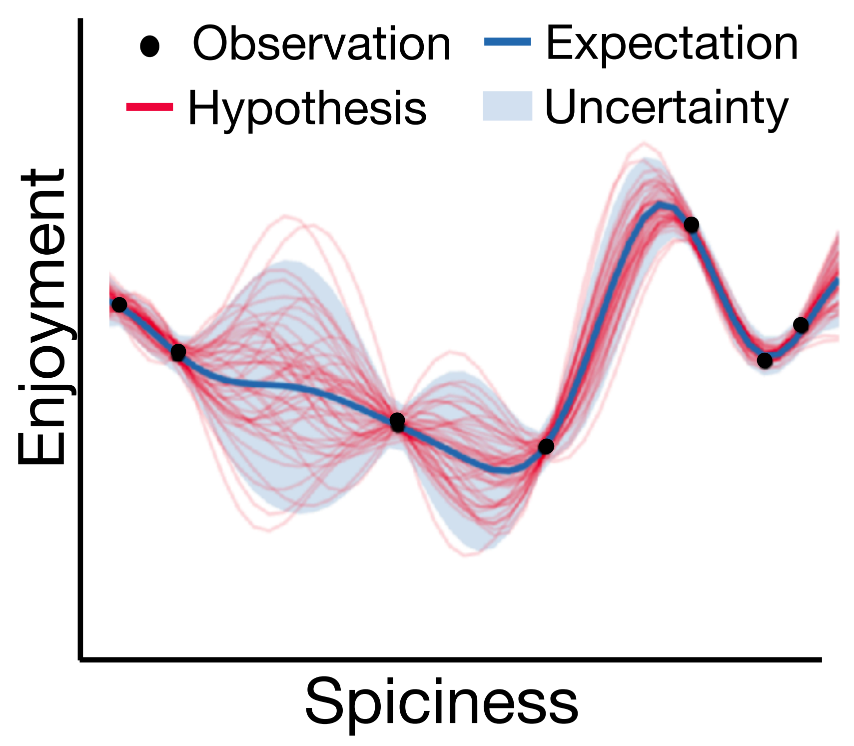


Wu et al., (*CBB* 2021)

Similarity can also capture relational structure

- The RBF kernel, like most classic accounts, represent similarity as distance in feature space
  - Learns smooth functions in a continuous domain

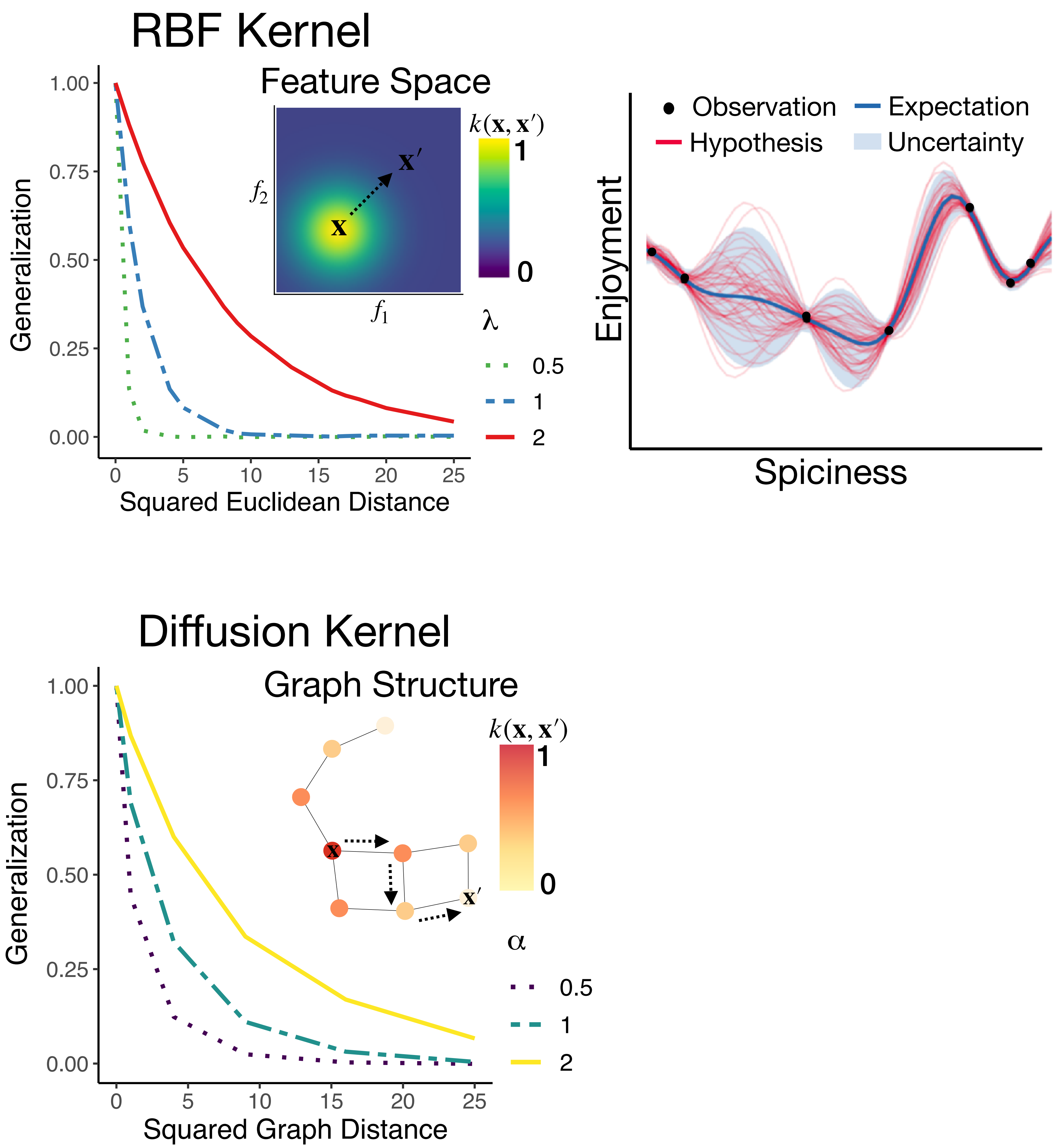




Wu et al., (*CBB* 2021)

Similarity can also capture relational structure

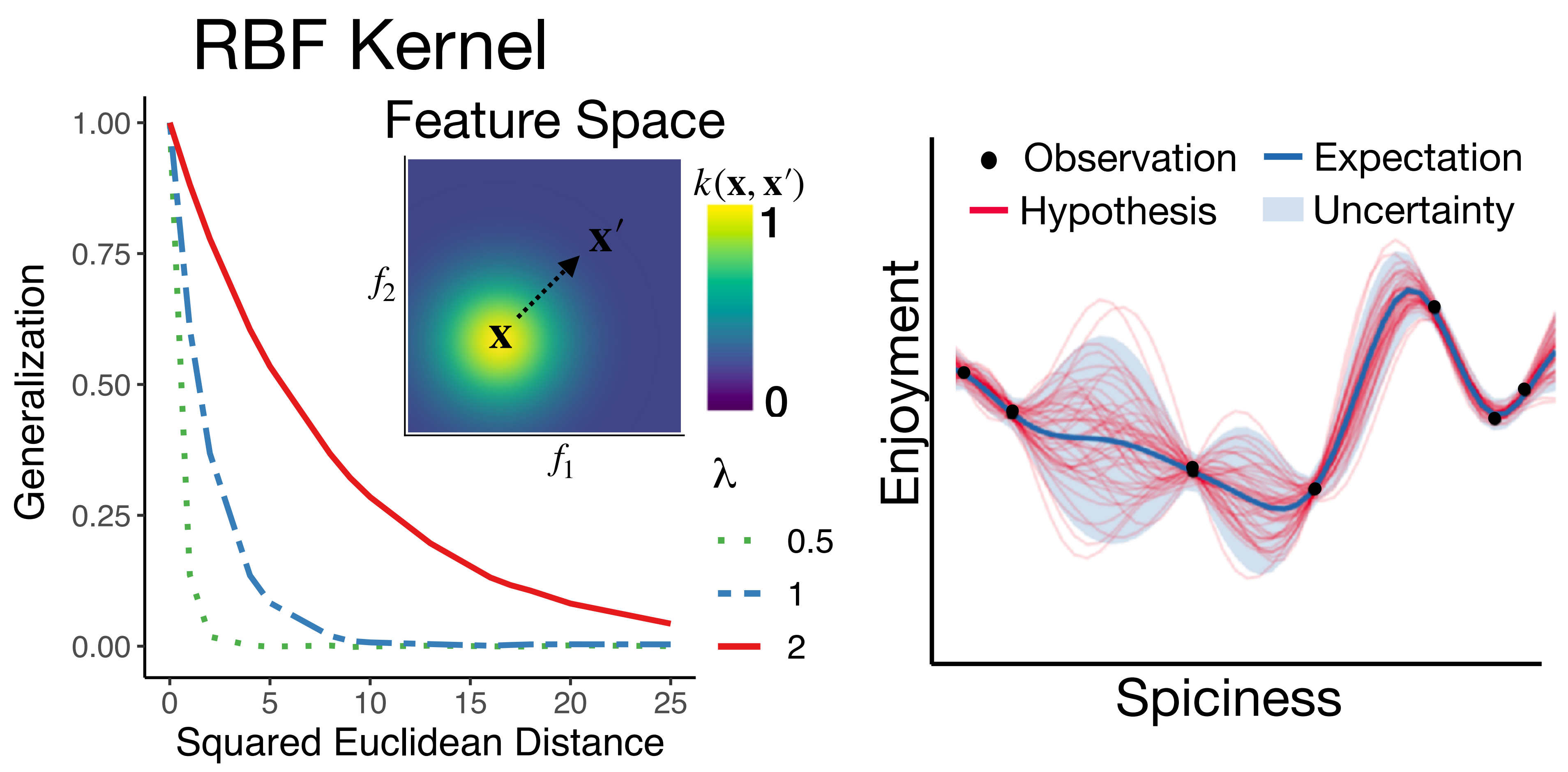
- The RBF kernel, like most classic accounts, represent similarity as distance in feature space
  - Learns smooth functions in a continuous domain
- A diffusion kernel represents similarity based on the connectivity of a graph
  - Learns functions on discrete graph representations

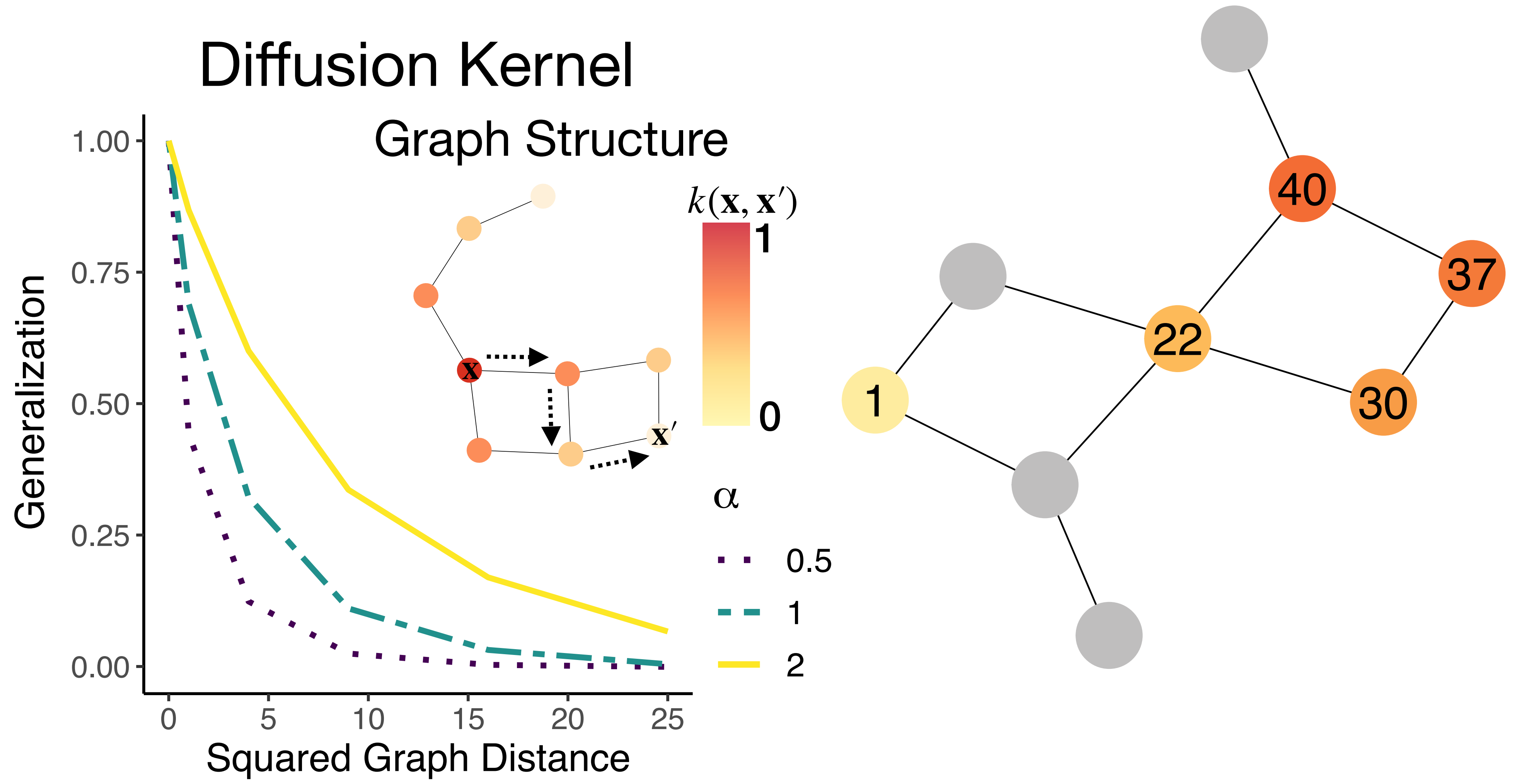


Wu et al., (*CBB* 2021)

Similarity can also capture relational structure

- The RBF kernel, like most classic accounts, represent similarity as distance in feature space
  - Learns smooth functions in a continuous domain
- A diffusion kernel represents similarity based on the connectivity of a graph
  - Learns functions on discrete graph representations

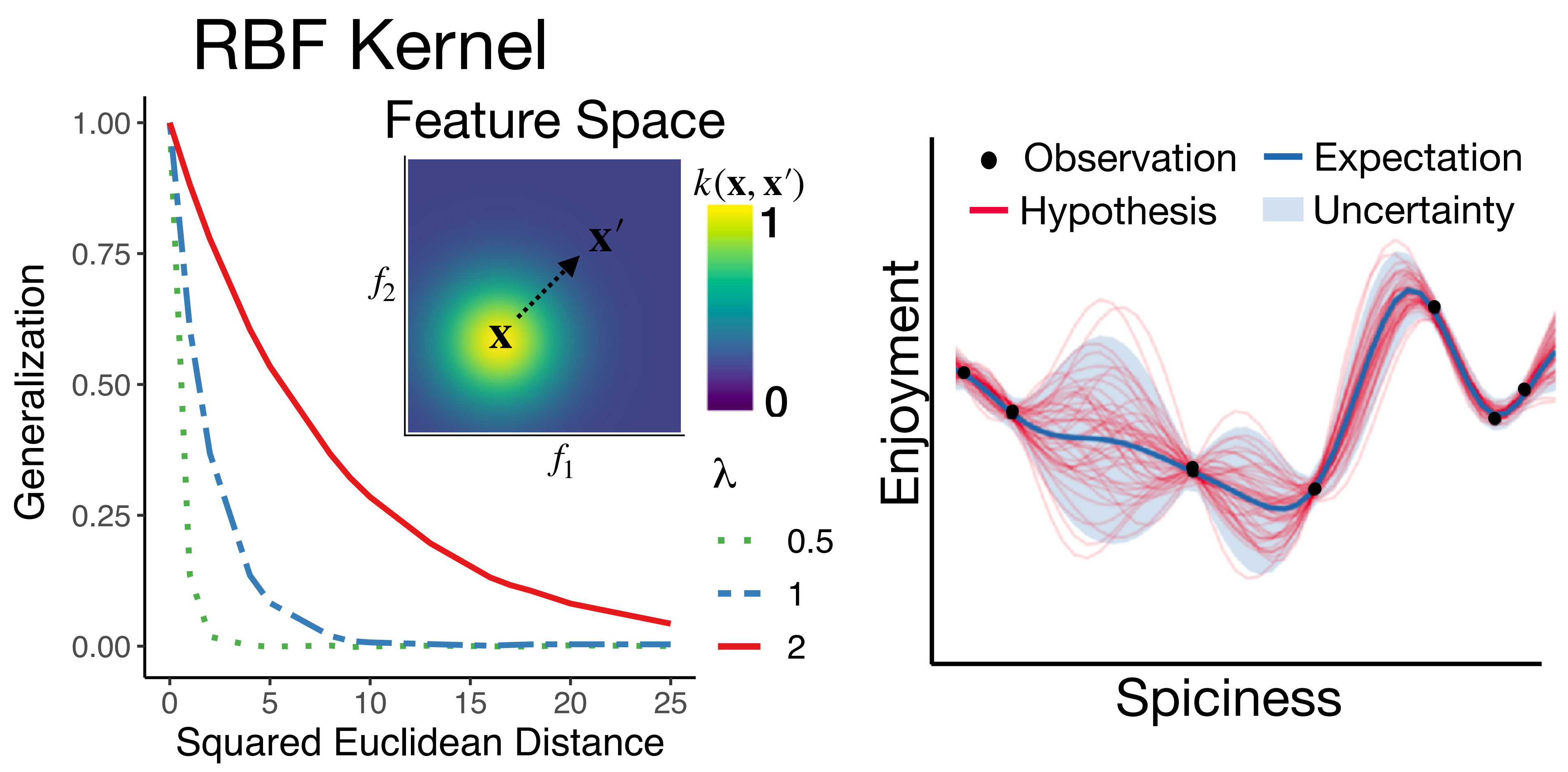


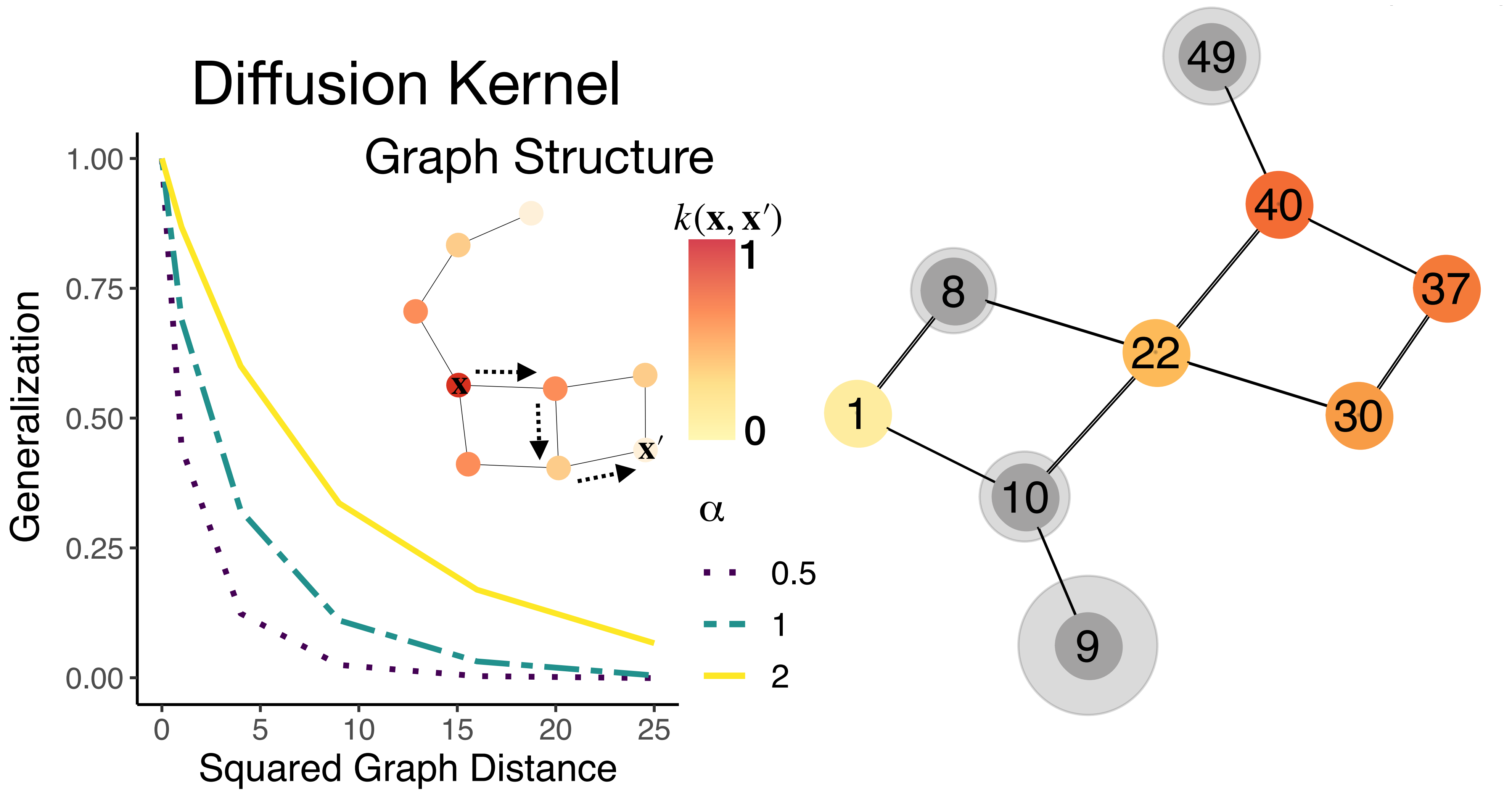


Wu et al., (*CBB* 2021)

Similarity can also capture relational structure

- The RBF kernel, like most classic accounts, represent similarity as distance in feature space
    - Learns smooth functions in a continuous domain
- A diffusion kernel represents similarity based on the connectivity of a graph
    - Learns functions on discrete graph representations

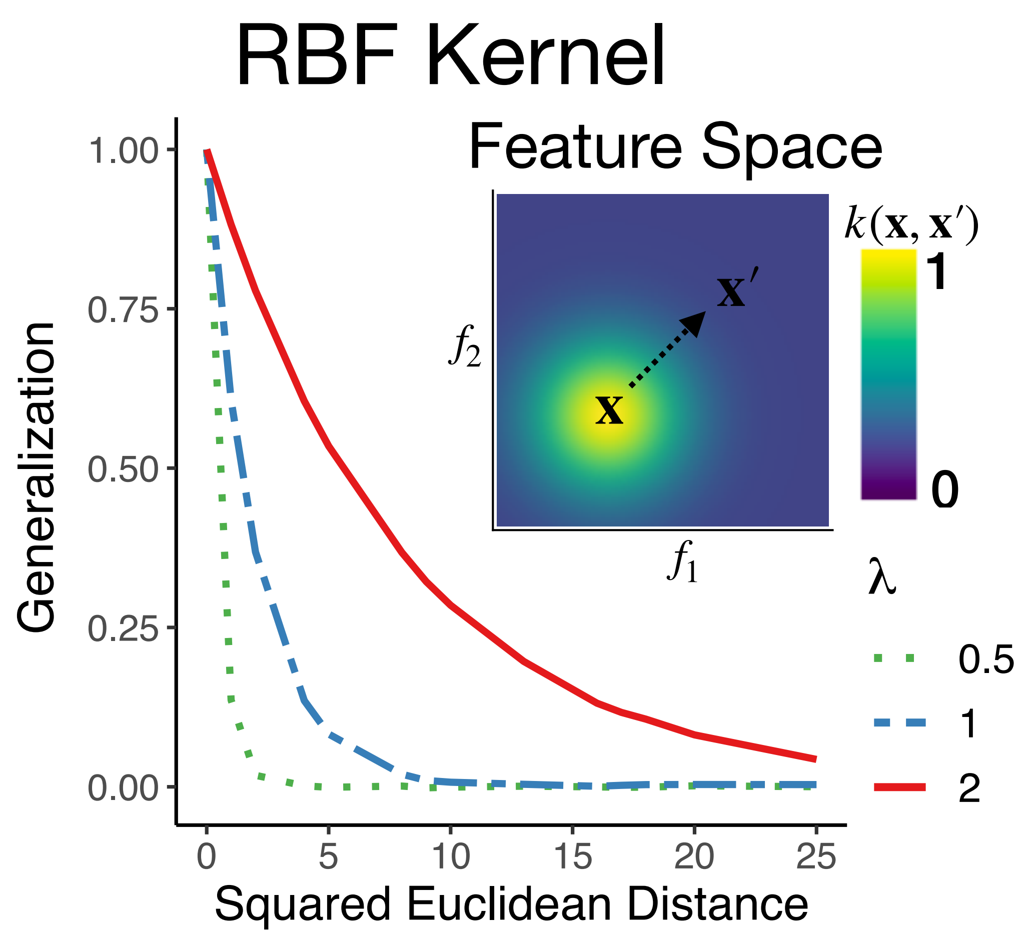


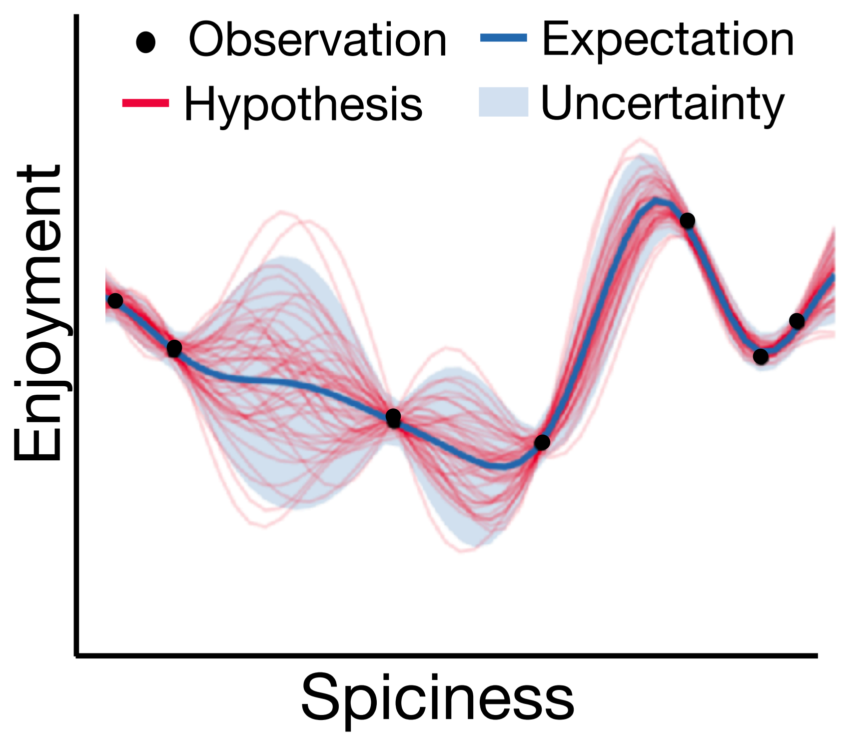


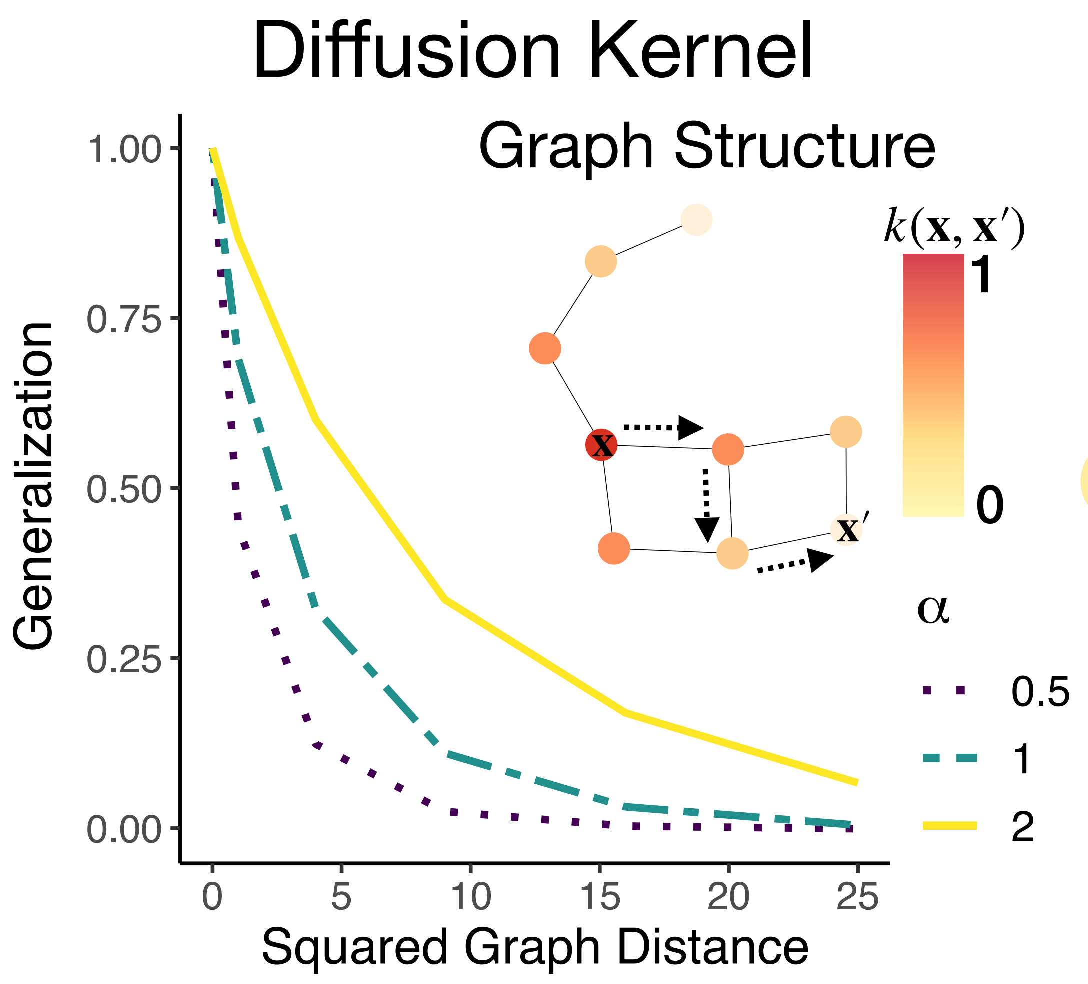
Wu et al., (*CBB* 2021)

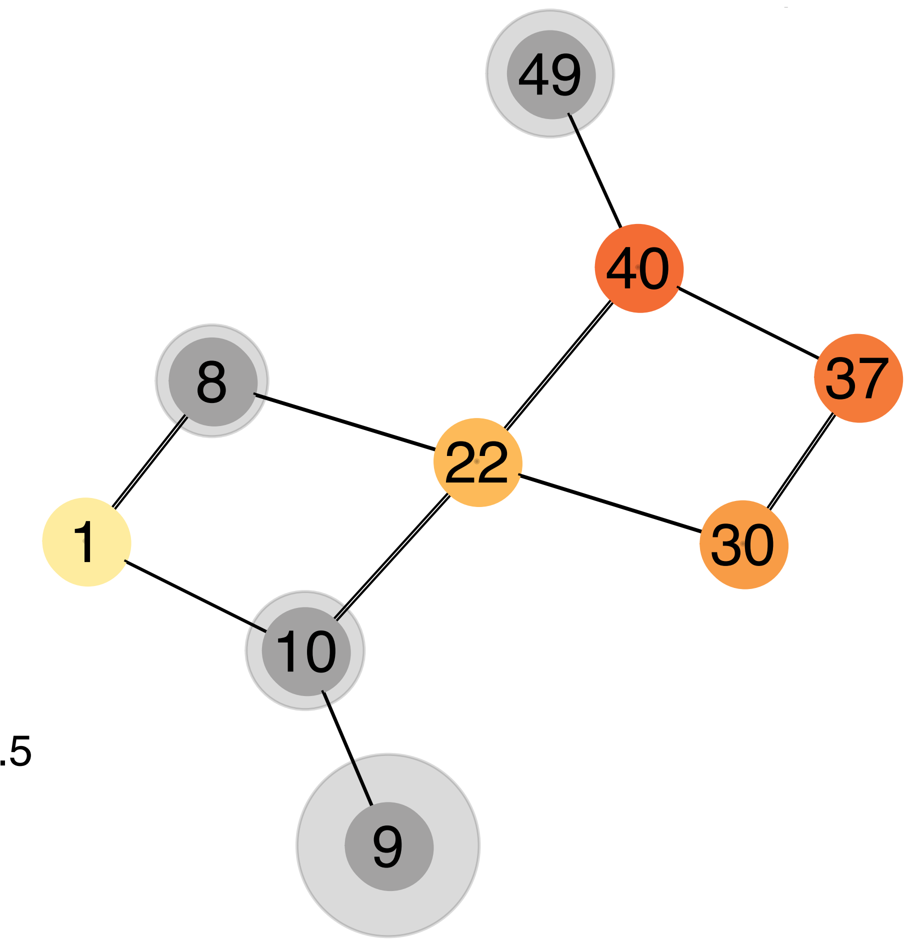
Similarity can also capture relational structure

- The RBF kernel, like most classic accounts, represent similarity as distance in feature space
    - Learns smooth functions in a continuous domain
- A diffusion kernel represents similarity based on the connectivity of a graph
    - Learns functions on discrete graph representations
- RBF kernel = Diffusion kernel in the limit of an infinitely fine lattice graph



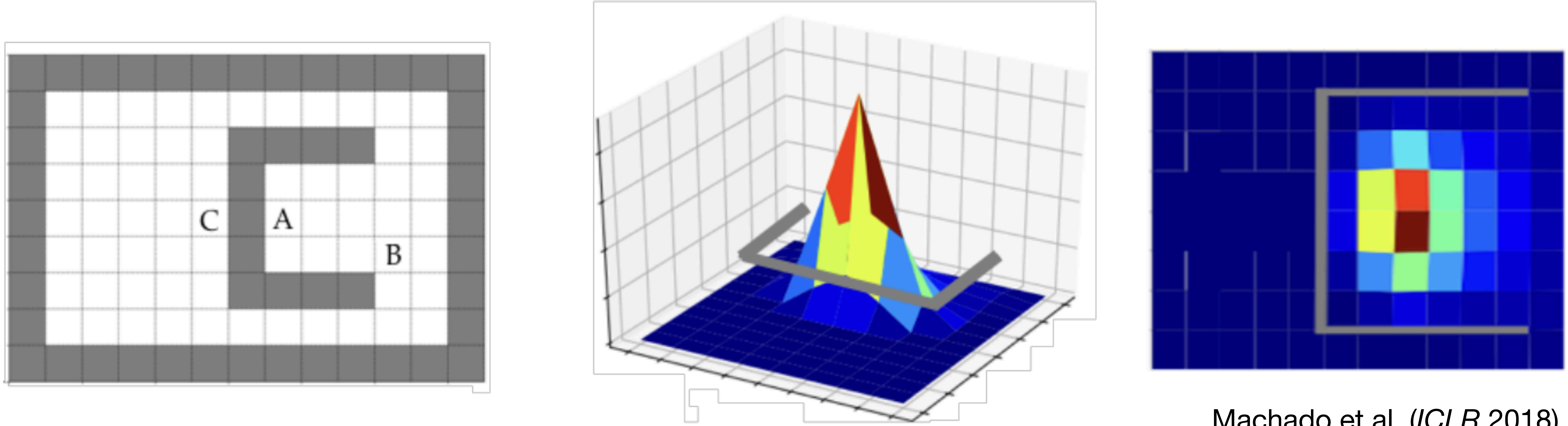






Wu et al., (*CBB* 2021)

Generalization based on transition dynamics



Machado et al. (*ICLR* 2018)

- A indicates a reward

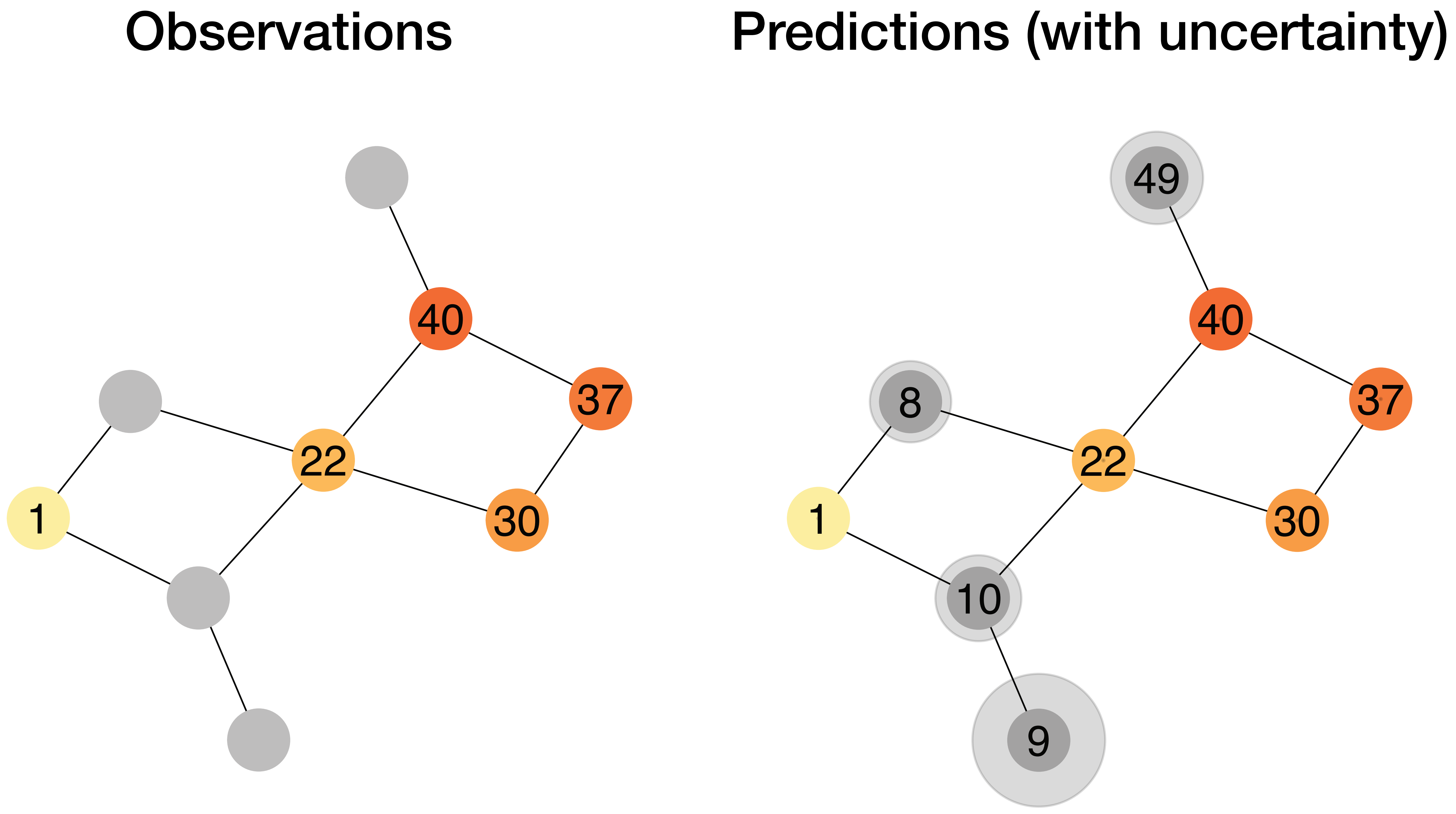
- Even though C is closer than B, the transition dynamics of the environment make it easier for B to reach A

Diffusion Kernel

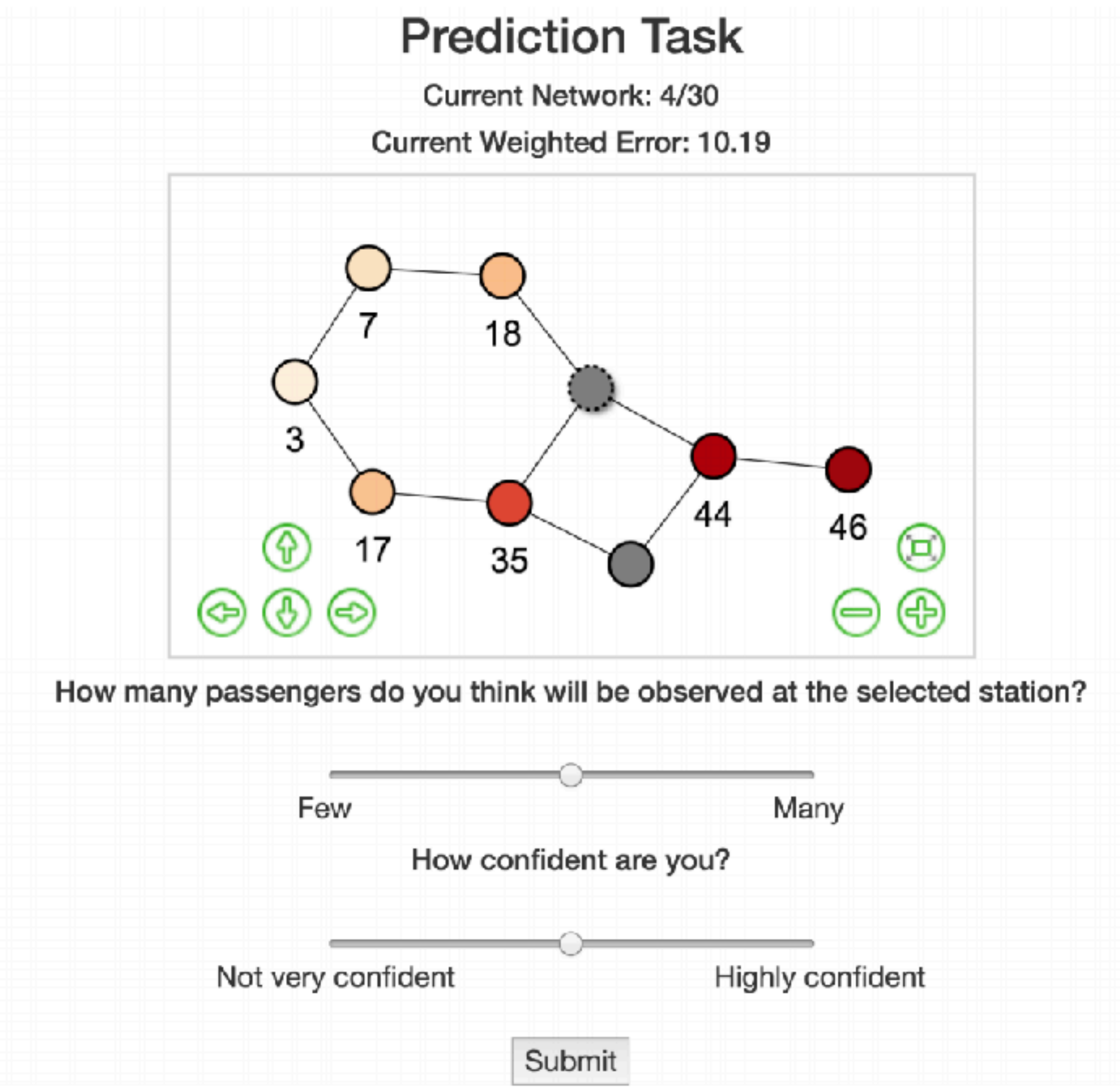
- Rather than similarity between features, we use the connectivity structure of the graph to define similarity

$$k_{DF}(s, s') = \exp(-\alpha L)$$

- Where L is the graph Laplacian
- α is a free parameter (diffusion level)
- The diffusion kernel assumes function values diffuse across the graph according to a random walk

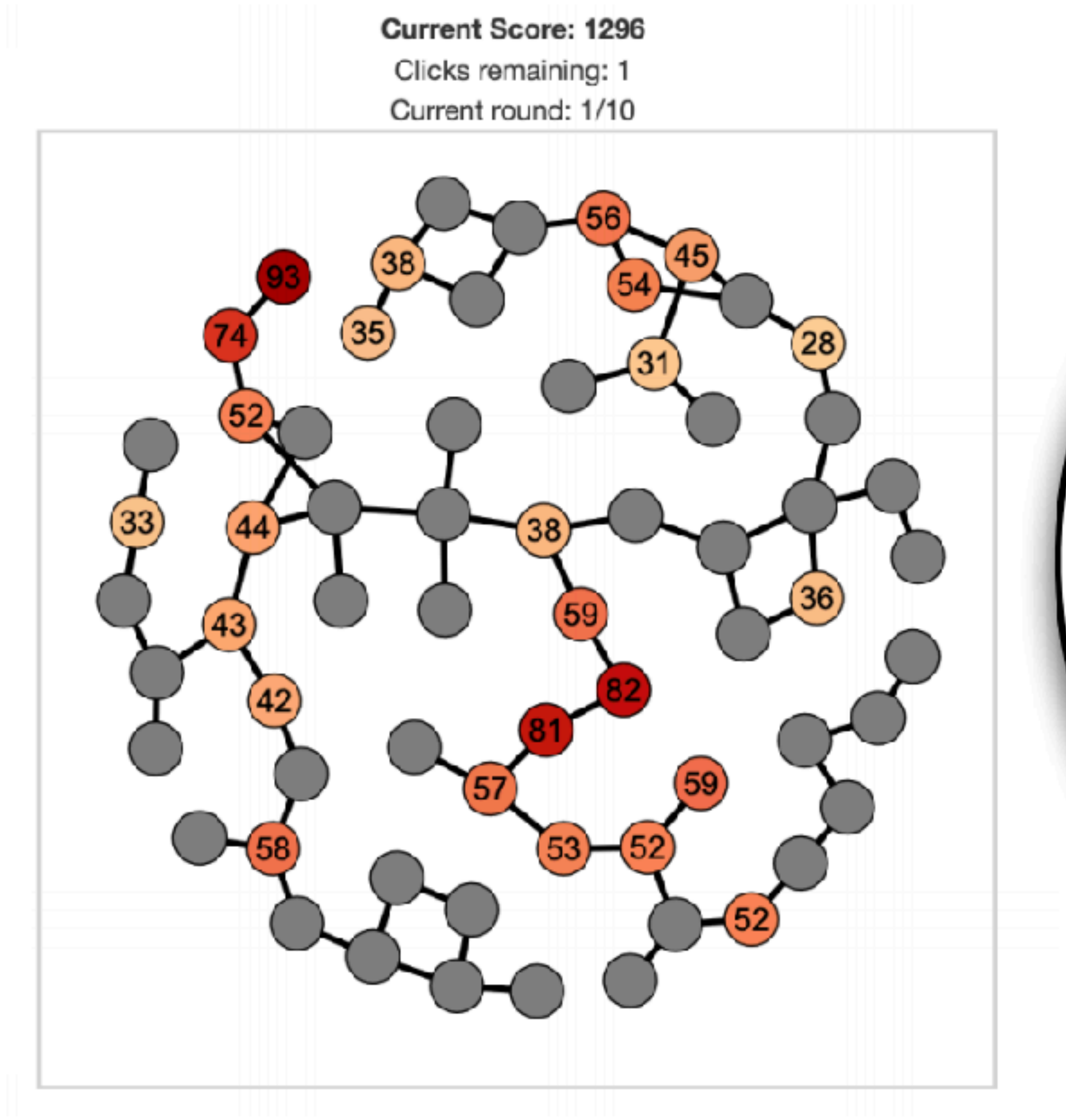


Experiment 1



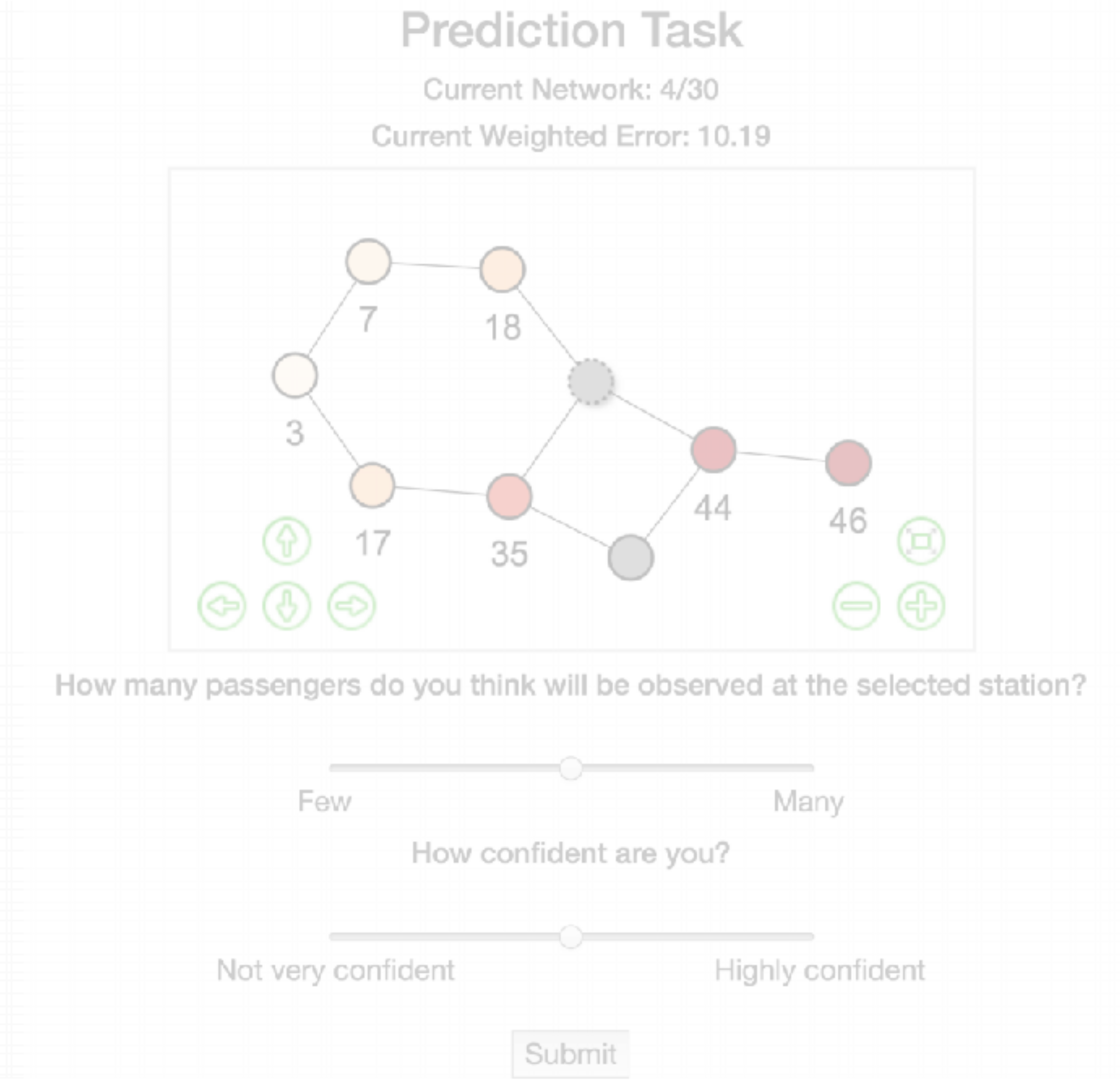
Experiment 2

Bandit Task



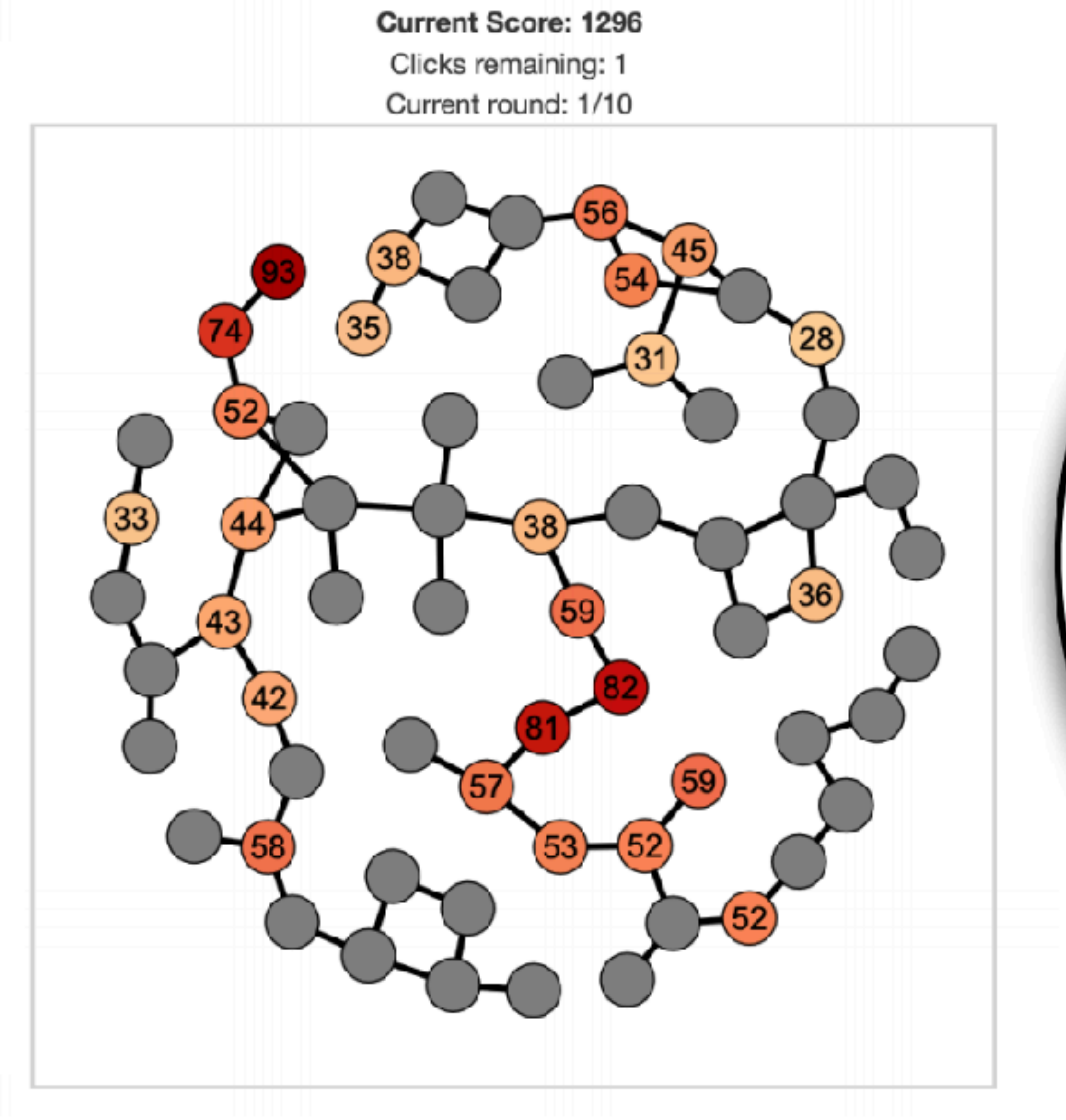
Bonus Round

Experiment 1



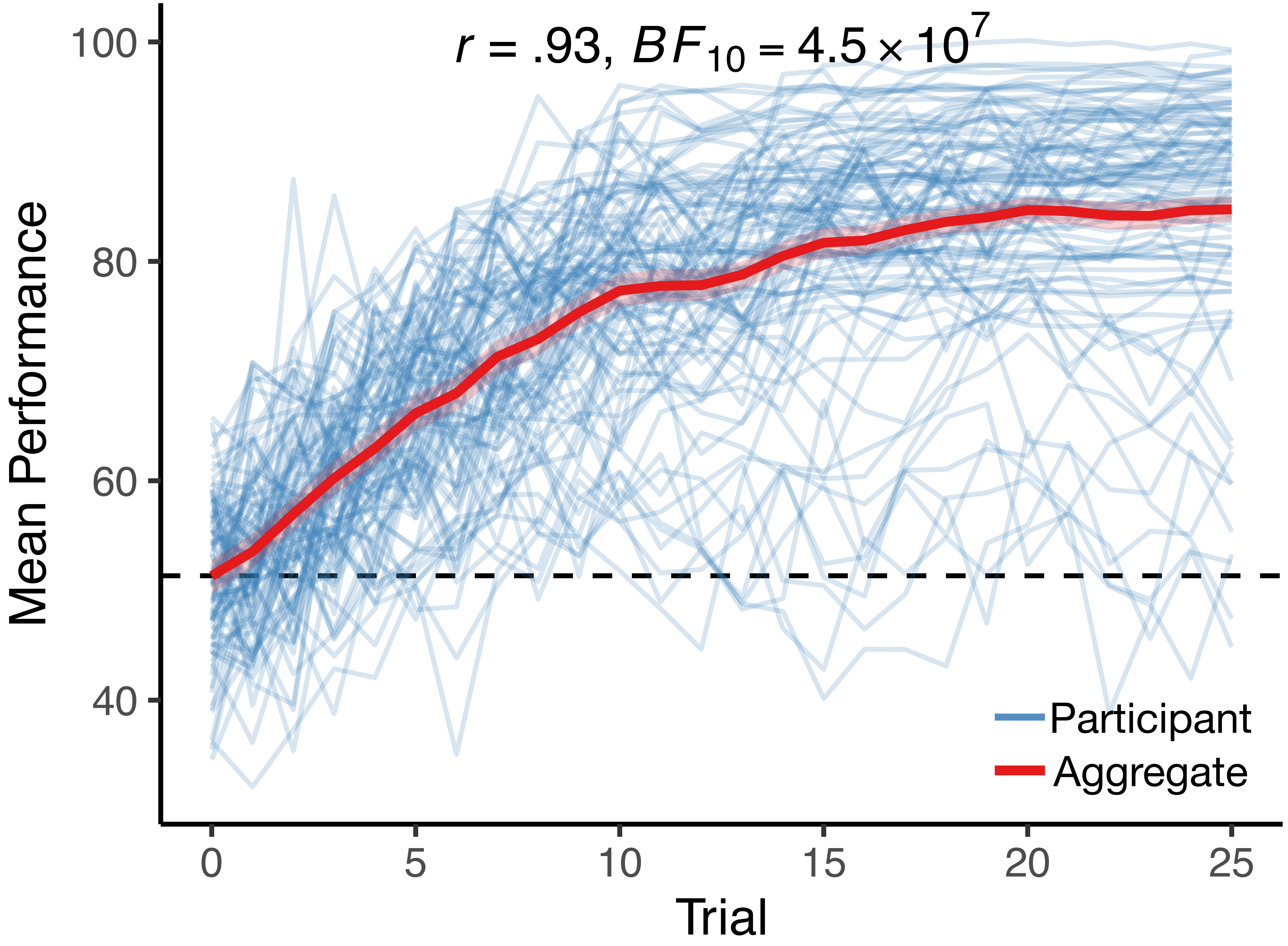
Experiment 2

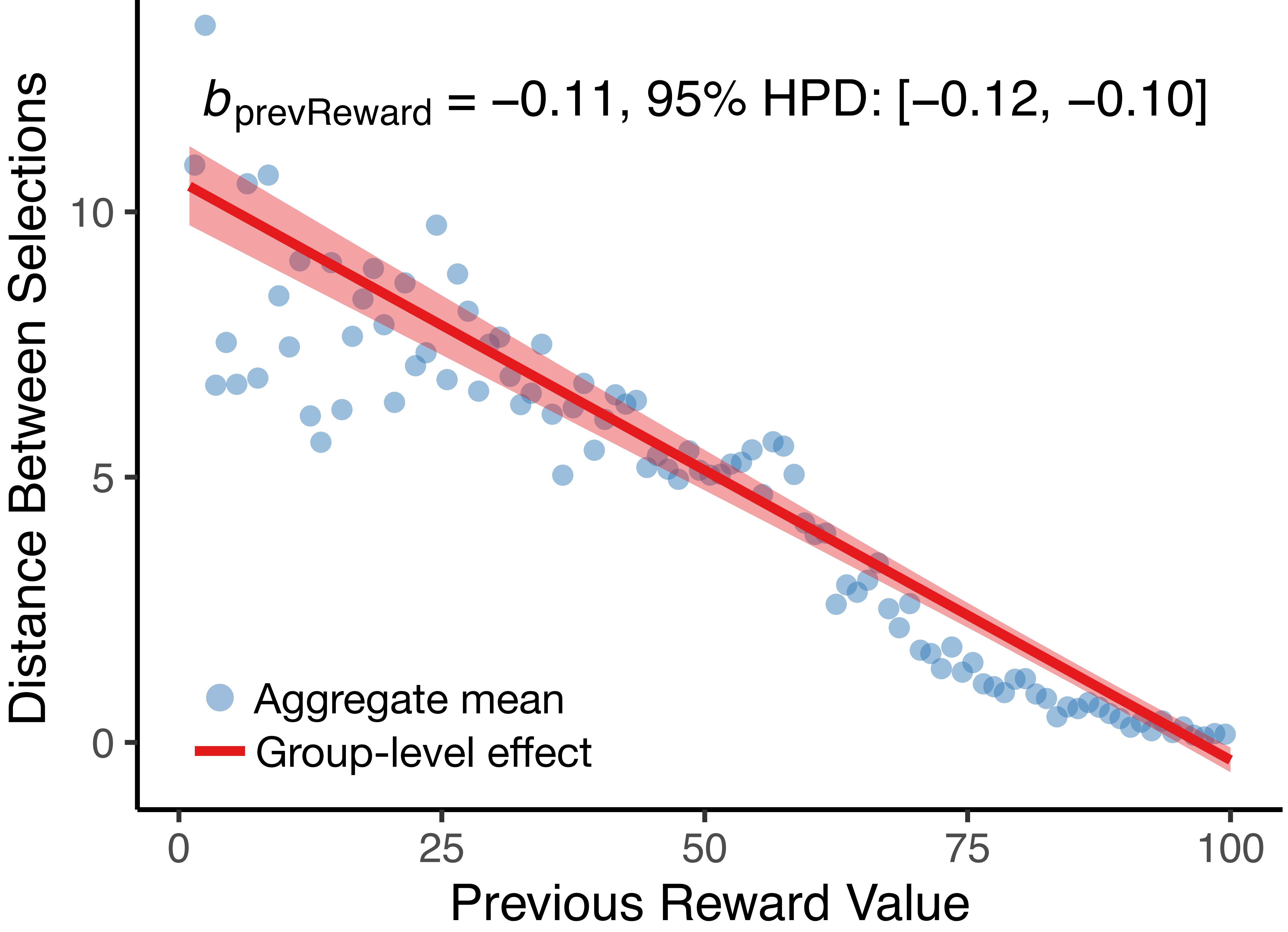
Bandit Task



Bonus Round

Behavioral Results

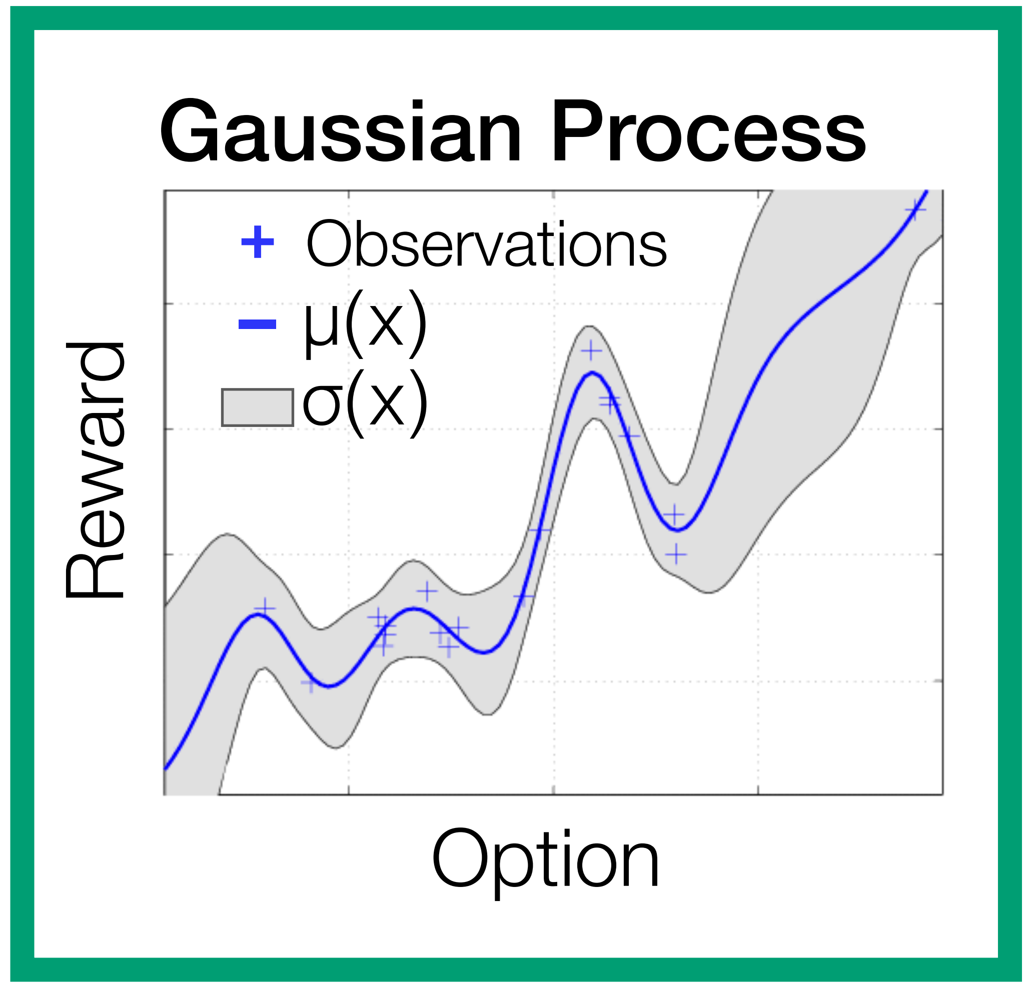




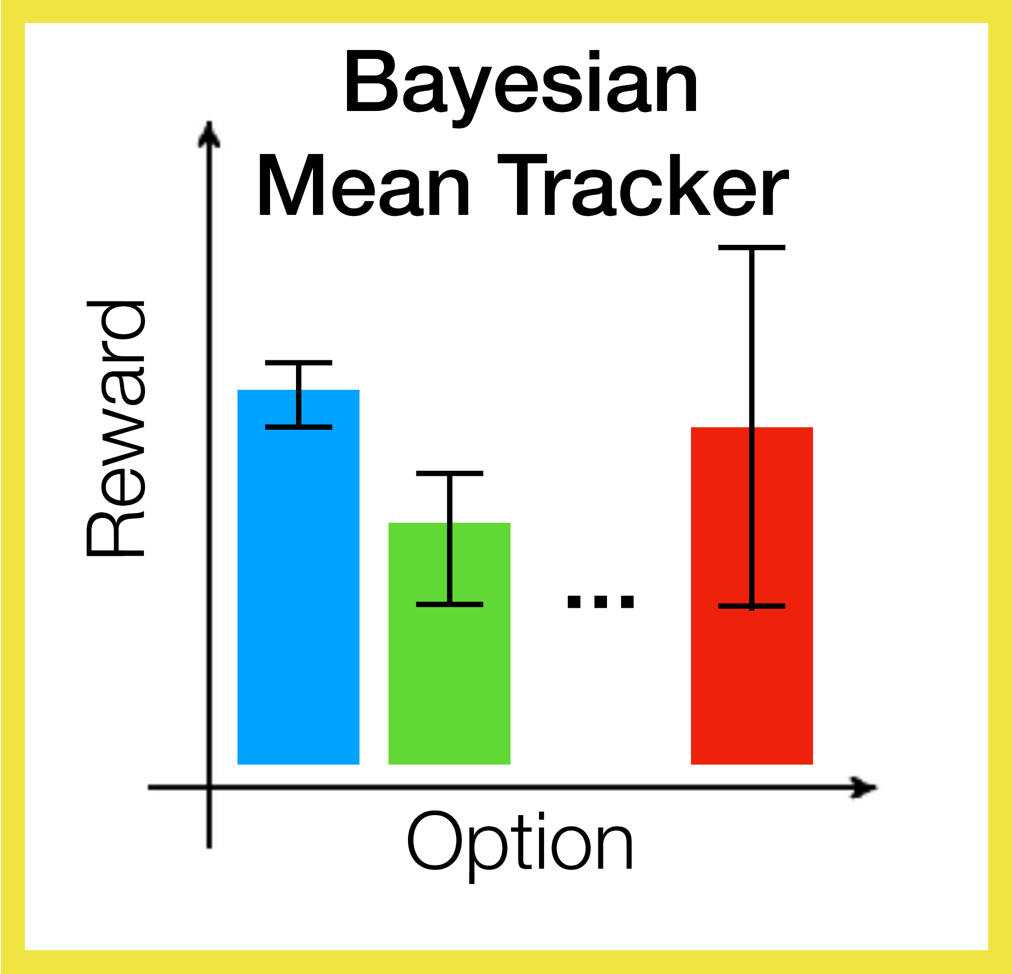
Wu, Schulz & Gershman (*CCN* 2019)

Model Results

Generalization



No generalization

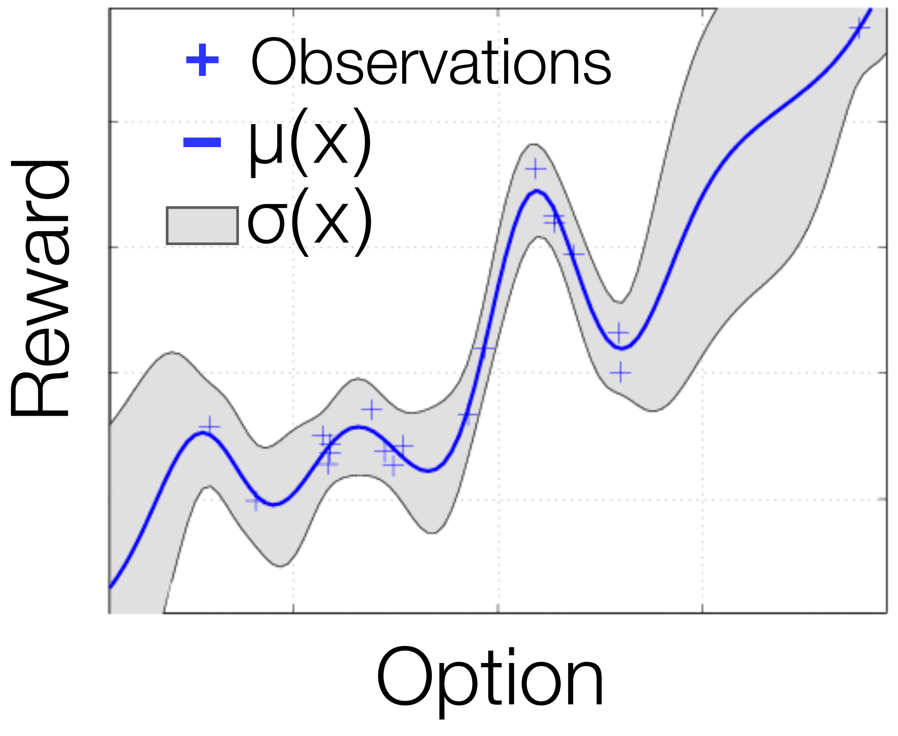


Model Results

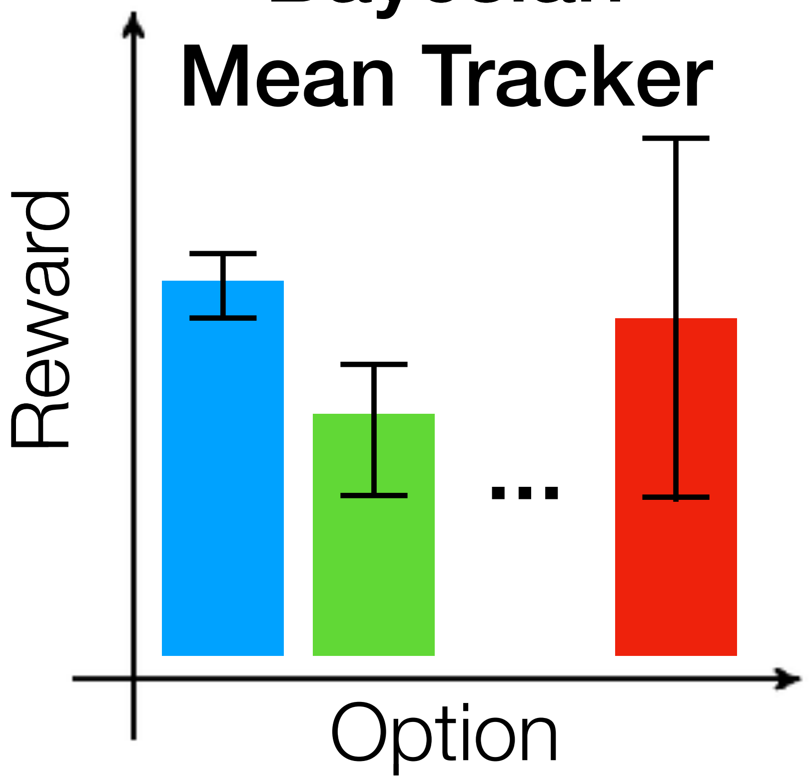
Generalization

No generalization

Gaussian Process

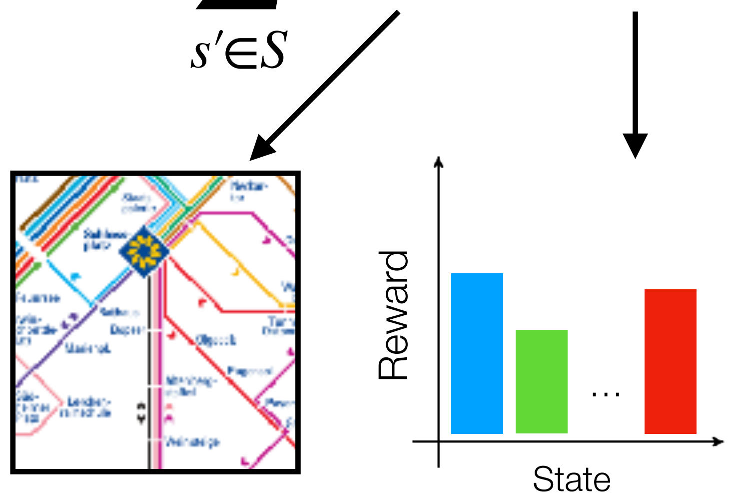


Bayesian Mean Tracker



Successor Representation

$$V^{\pi}(s,a) = \sum_{s' \in S} M(s,s',a)R(s')$$



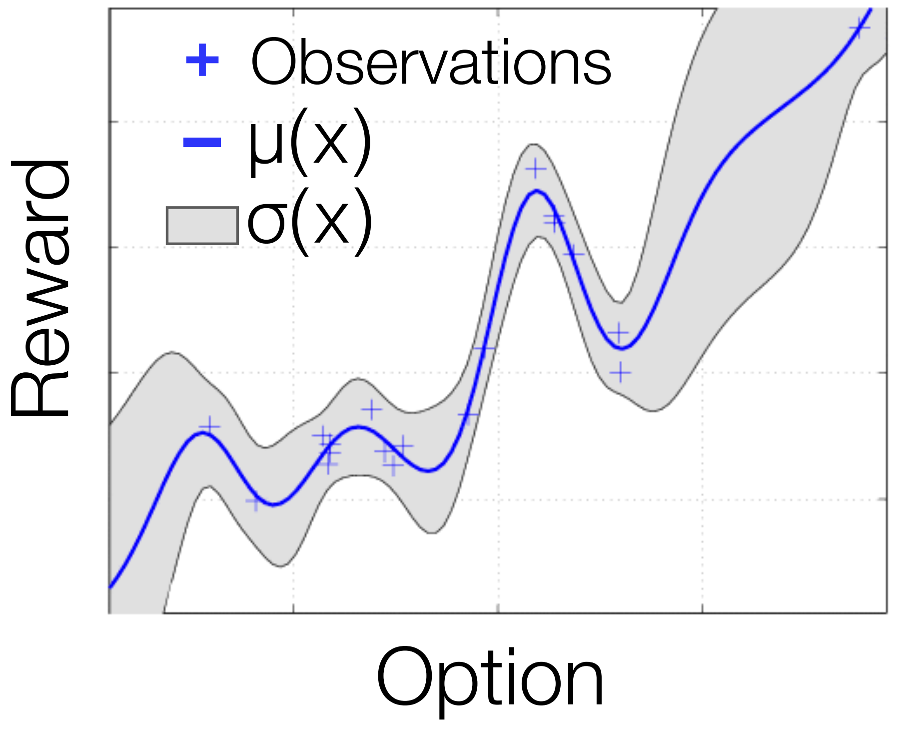
Wu, Schulz & Gershman (*CCN* 2019)

Model Results

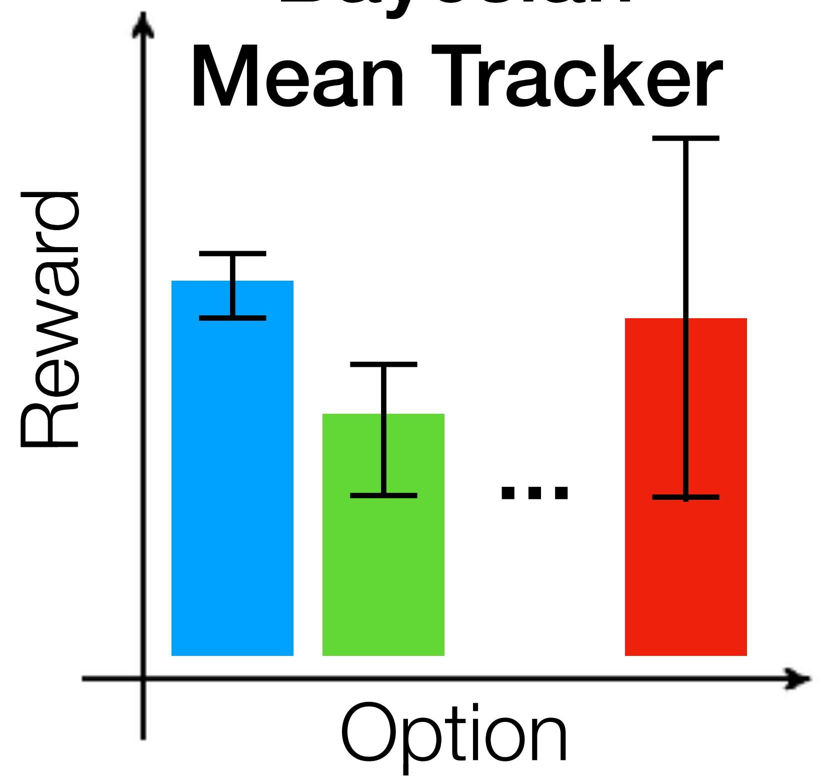
Generalization | **No generalization**

directed + random exploration

Gaussian Process



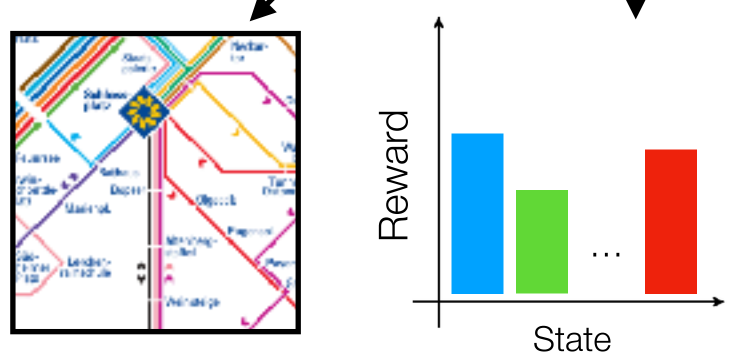
Bayesian Mean Tracker



random exploration

Successor Representation

$$V^{\pi}(s,a) = \sum_{s' \in S} M(s,s',a)R(s')$$



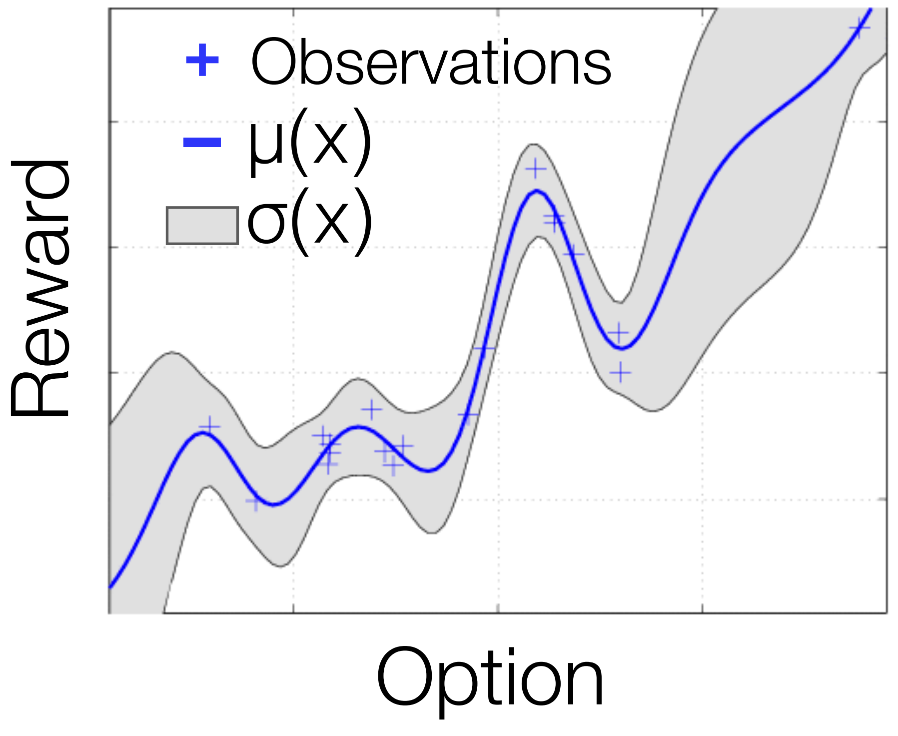
Model Results

directed + random exploration

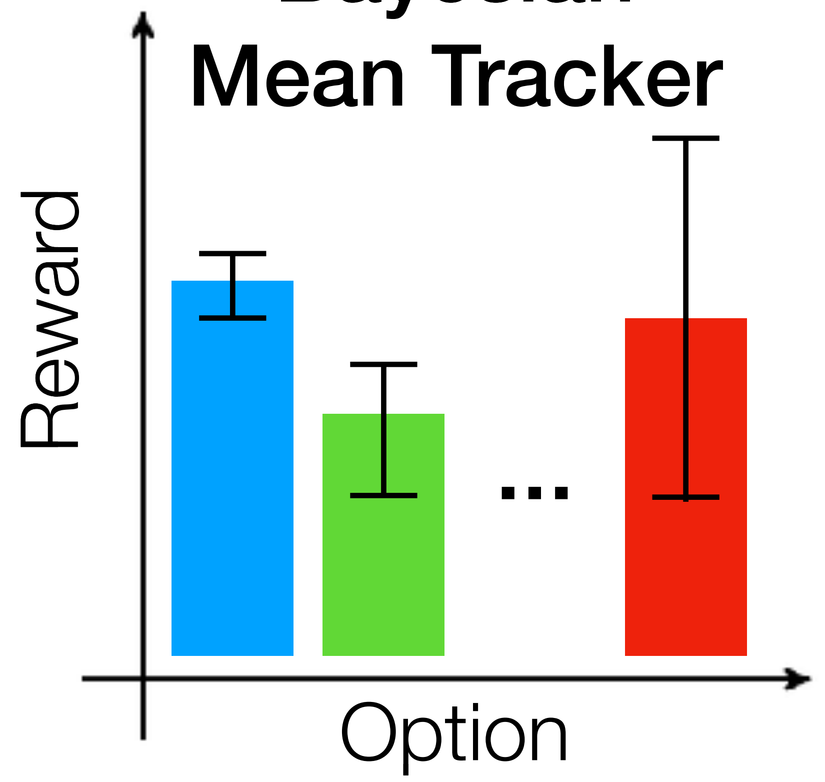
Generalization

No generalization

Gaussian Process



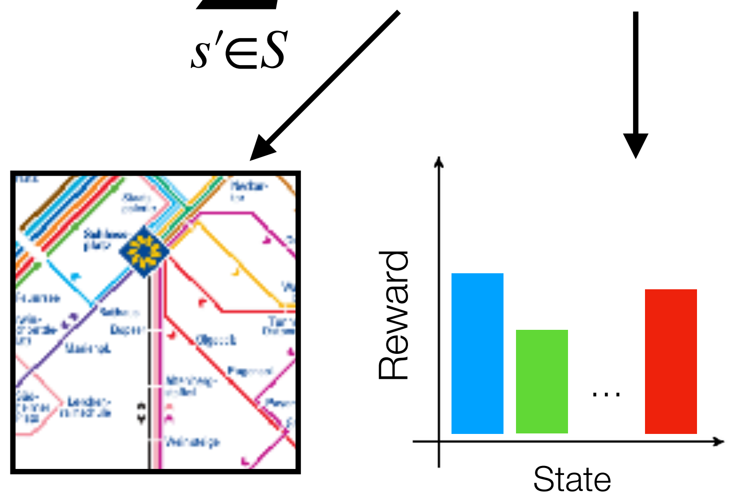
Bayesian Mean Tracker



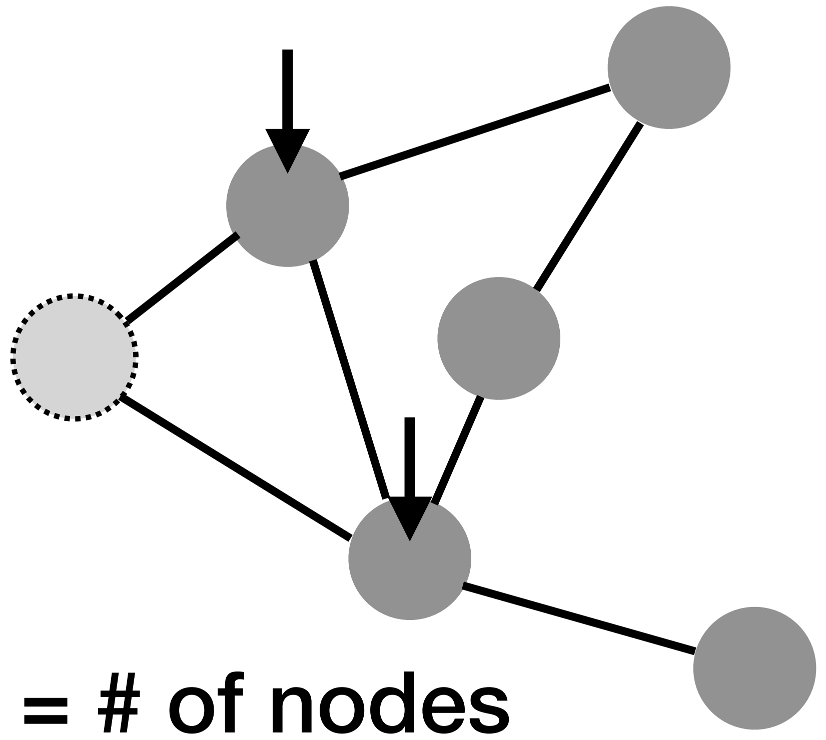
random exploration

Successor Representation

$$V^{\pi}(s,a) = \sum_{s' \in S} M(s,s',a)R(s')$$



***k*-Nearest Neighbors**



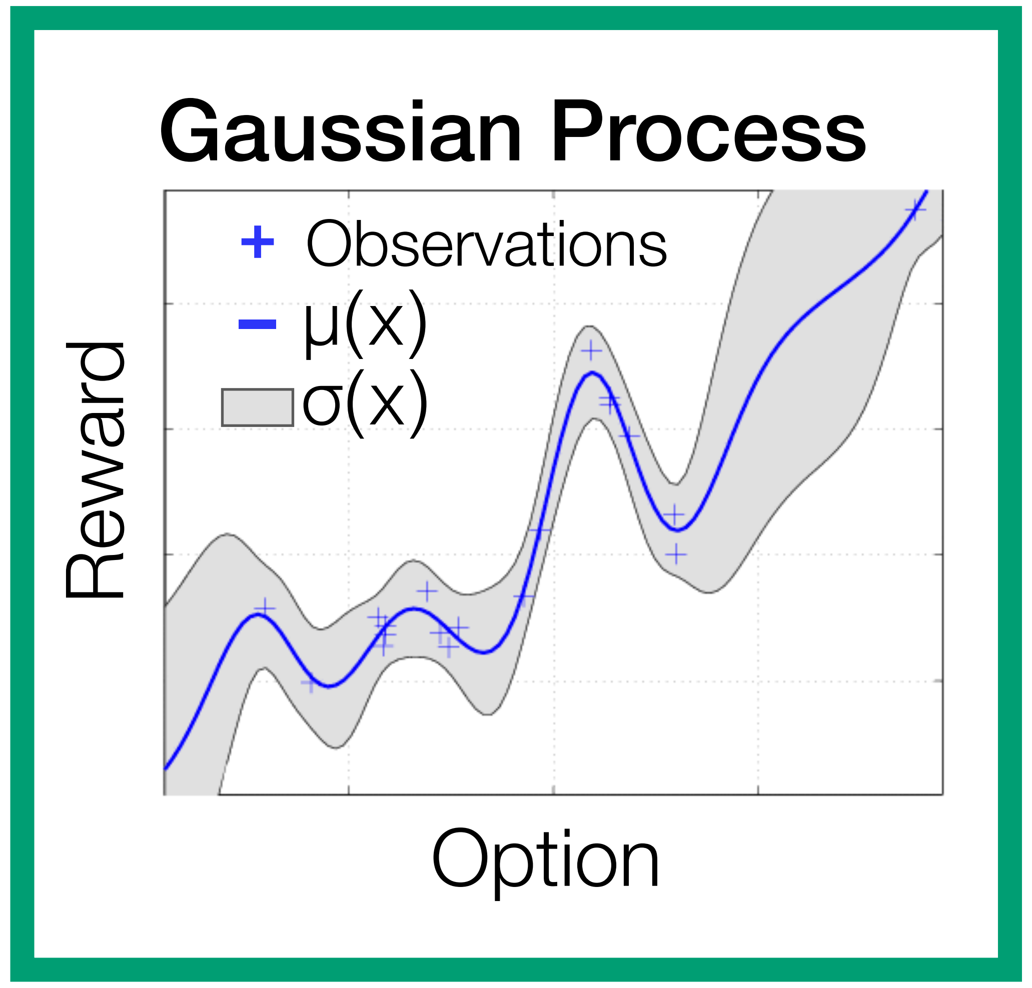
***k* = # of nodes**

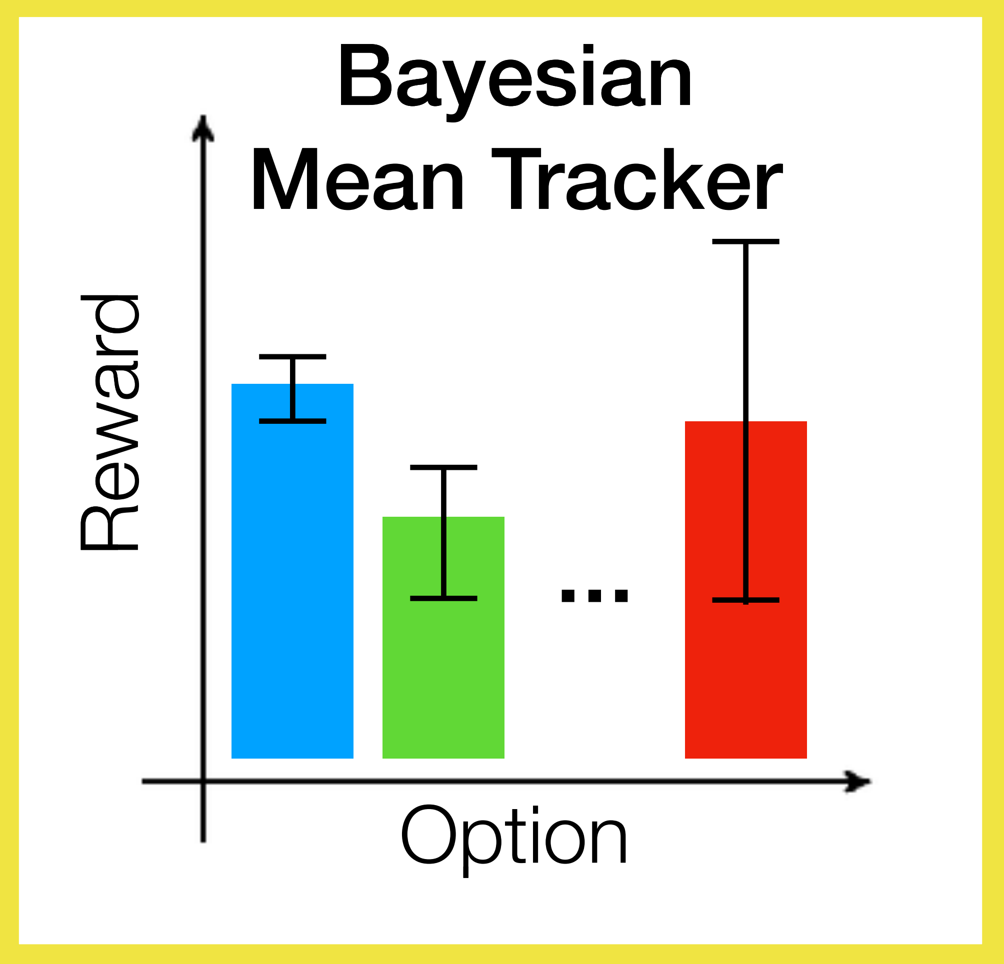
Wu, Schulz & Gershman (*CCN* 2019)

Model Results

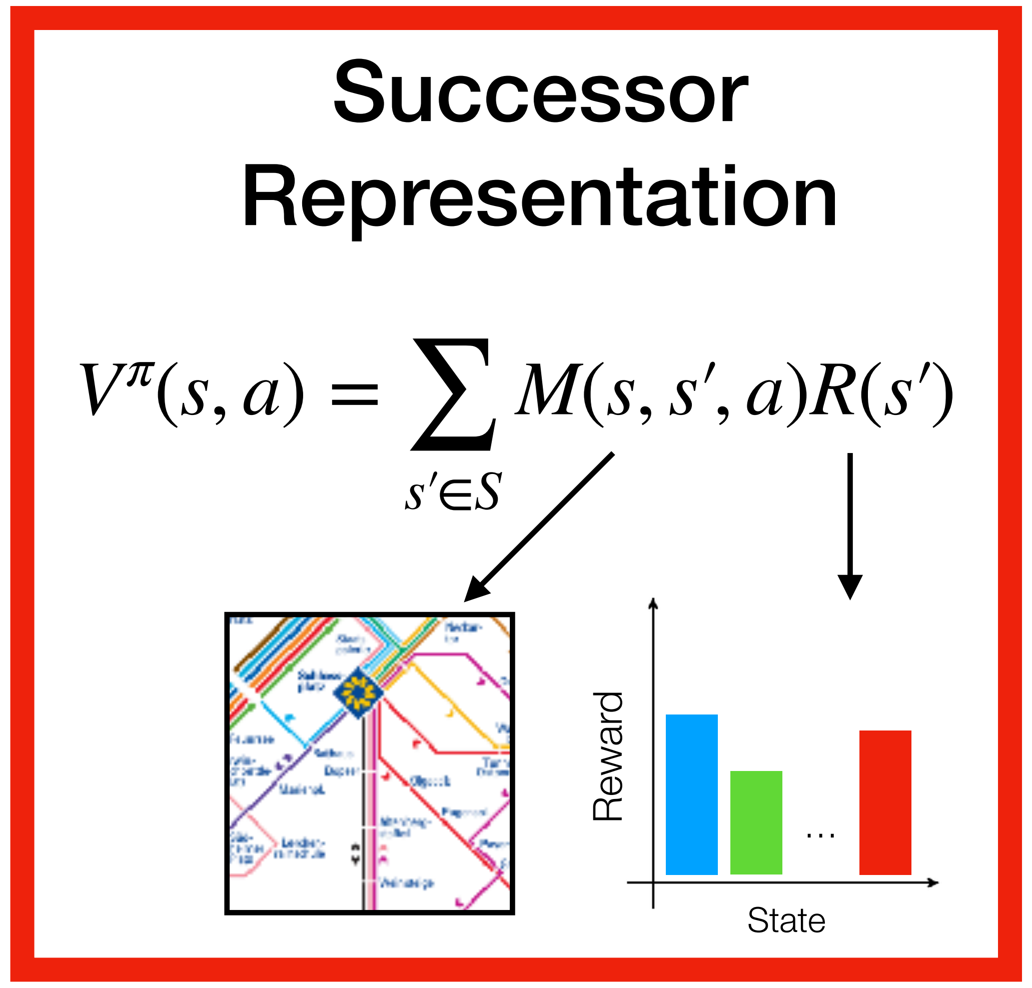
Generalization | **No generalization**

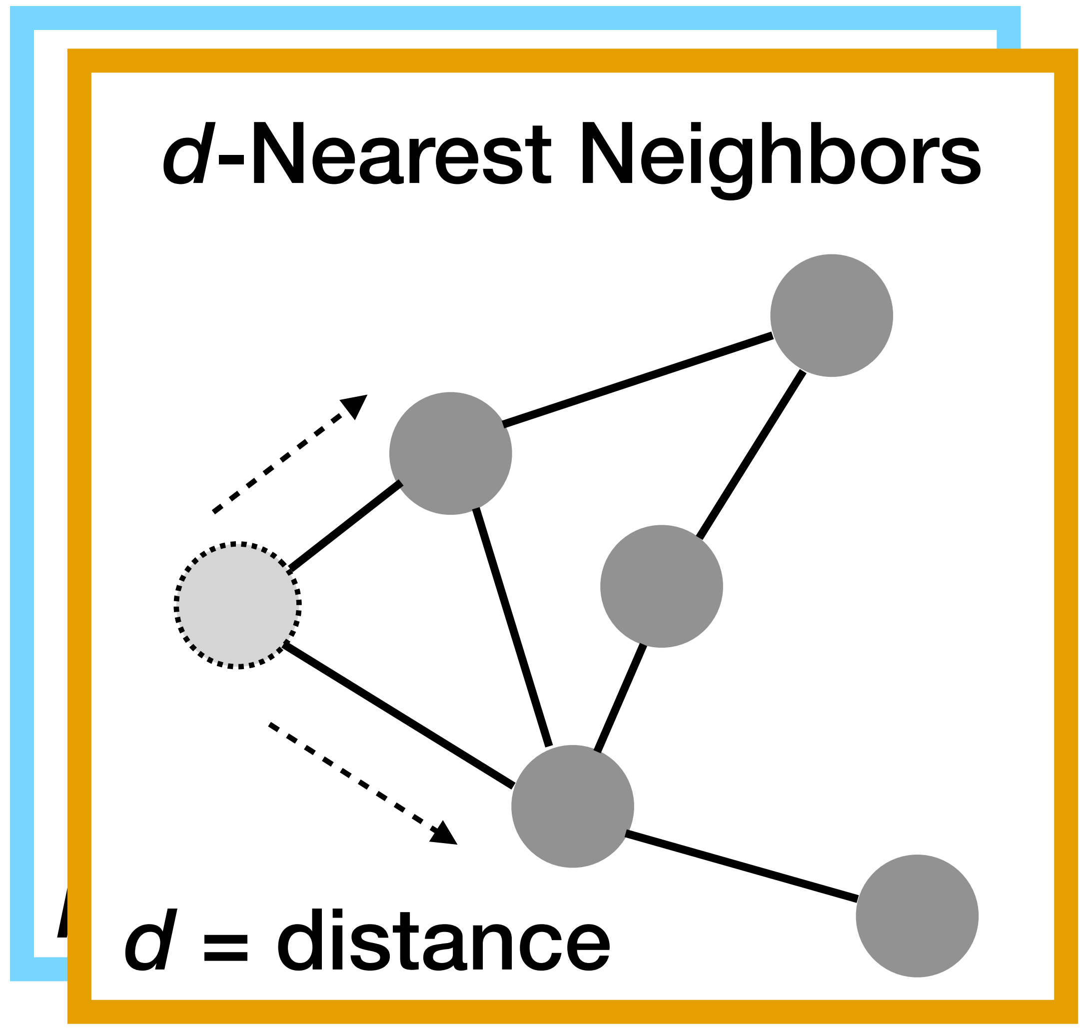
directed + random exploration





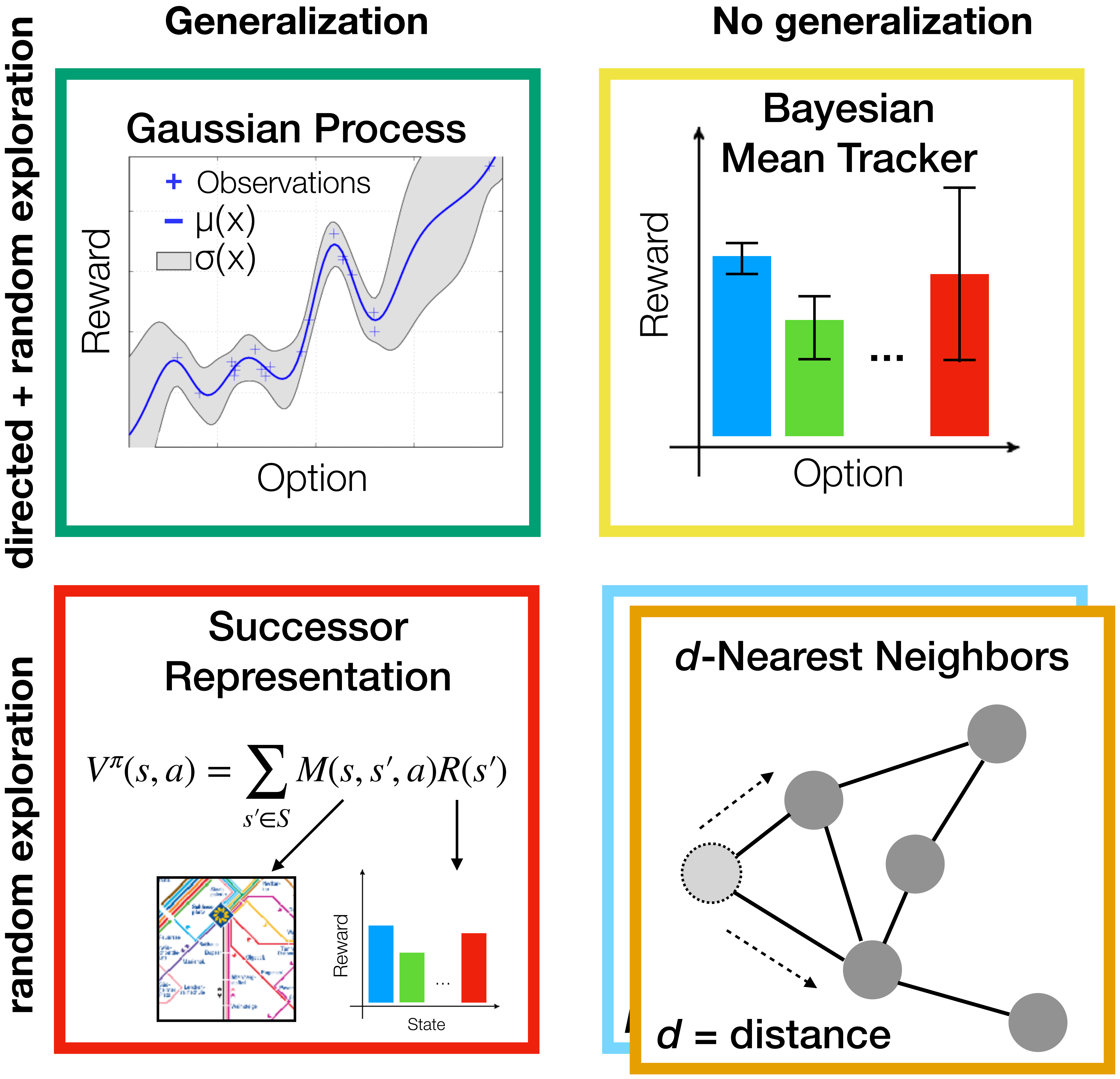
random exploration

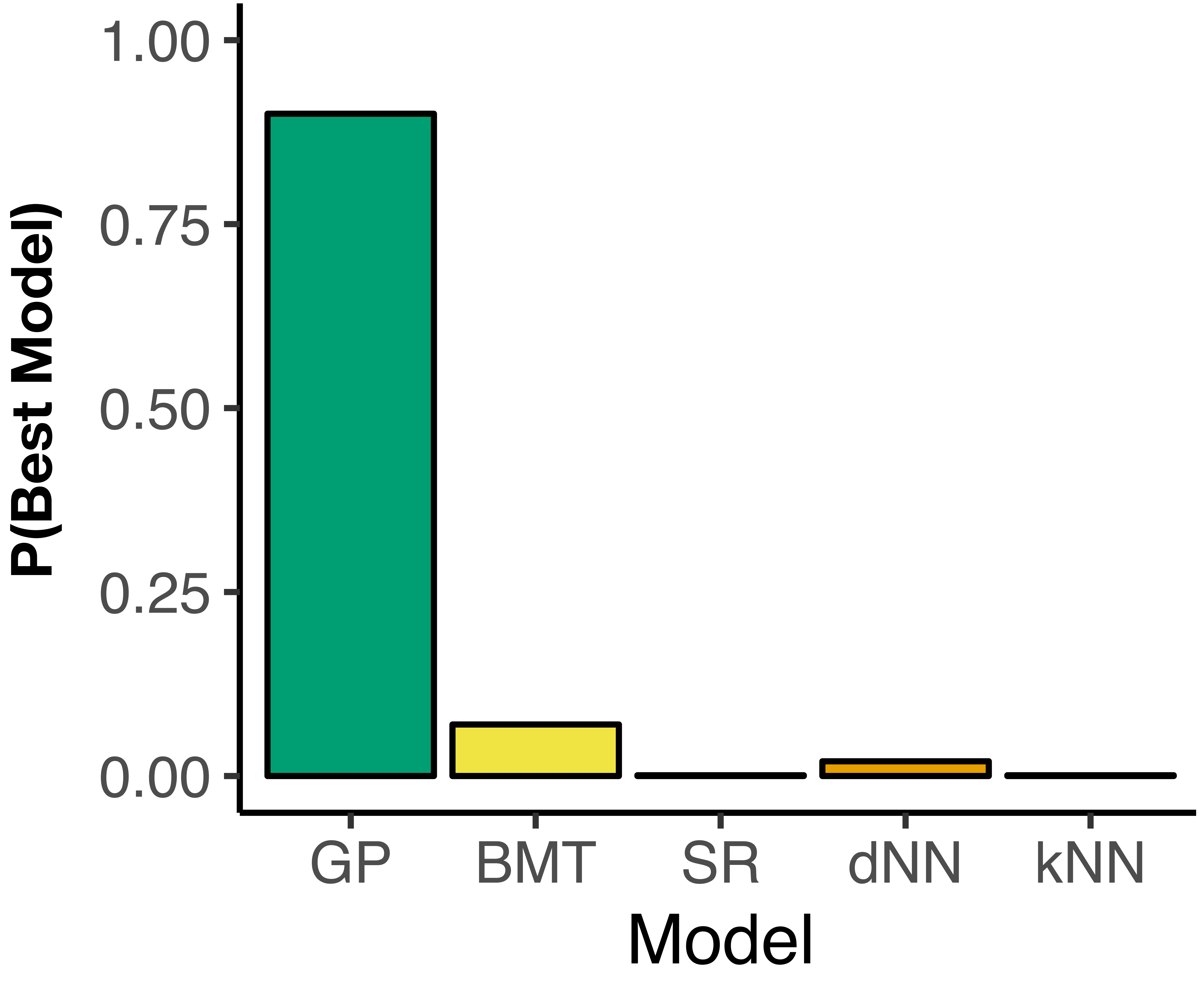




Wu, Schulz & Gershman (*CCN* 2019)

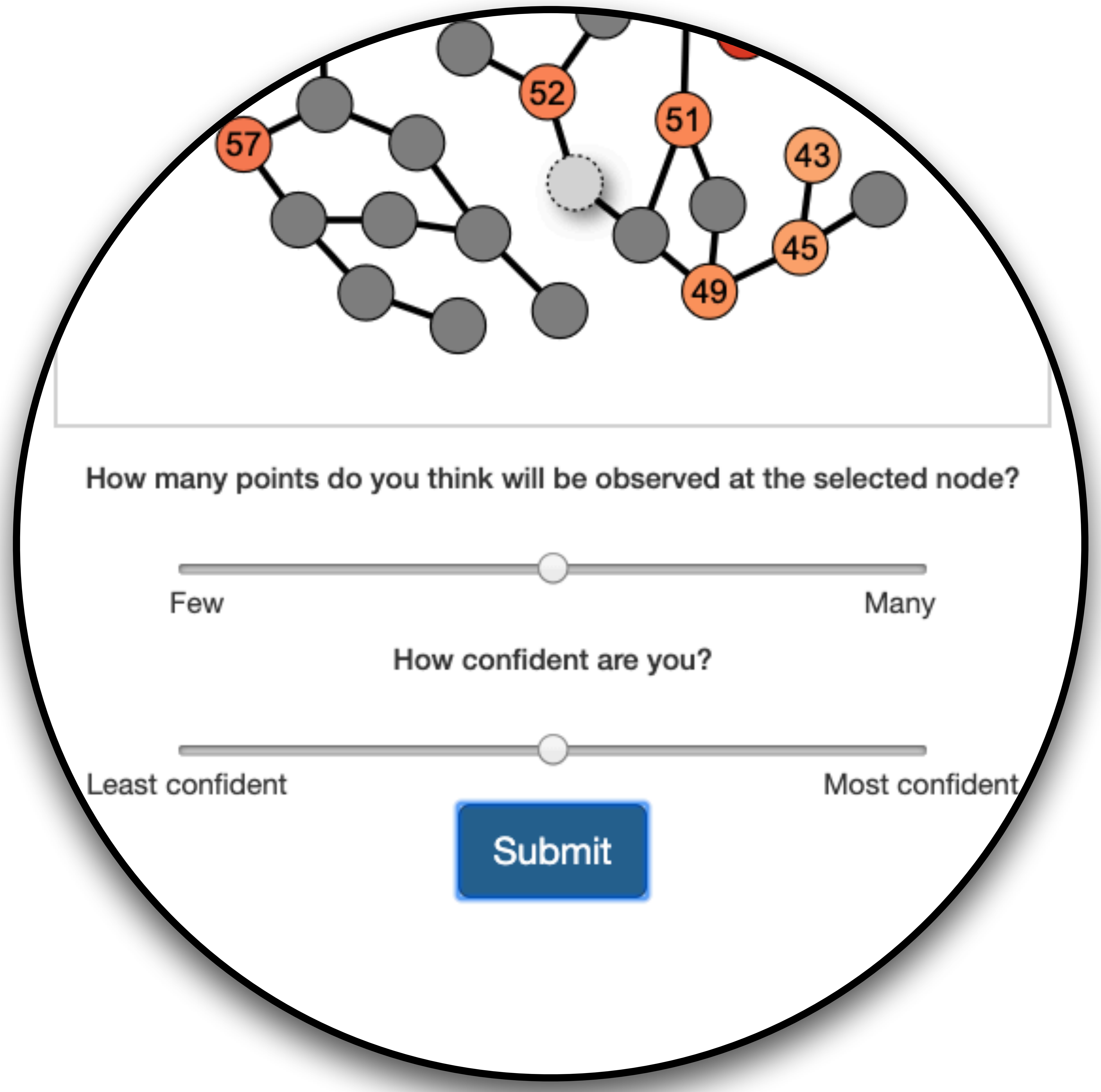
Model Results

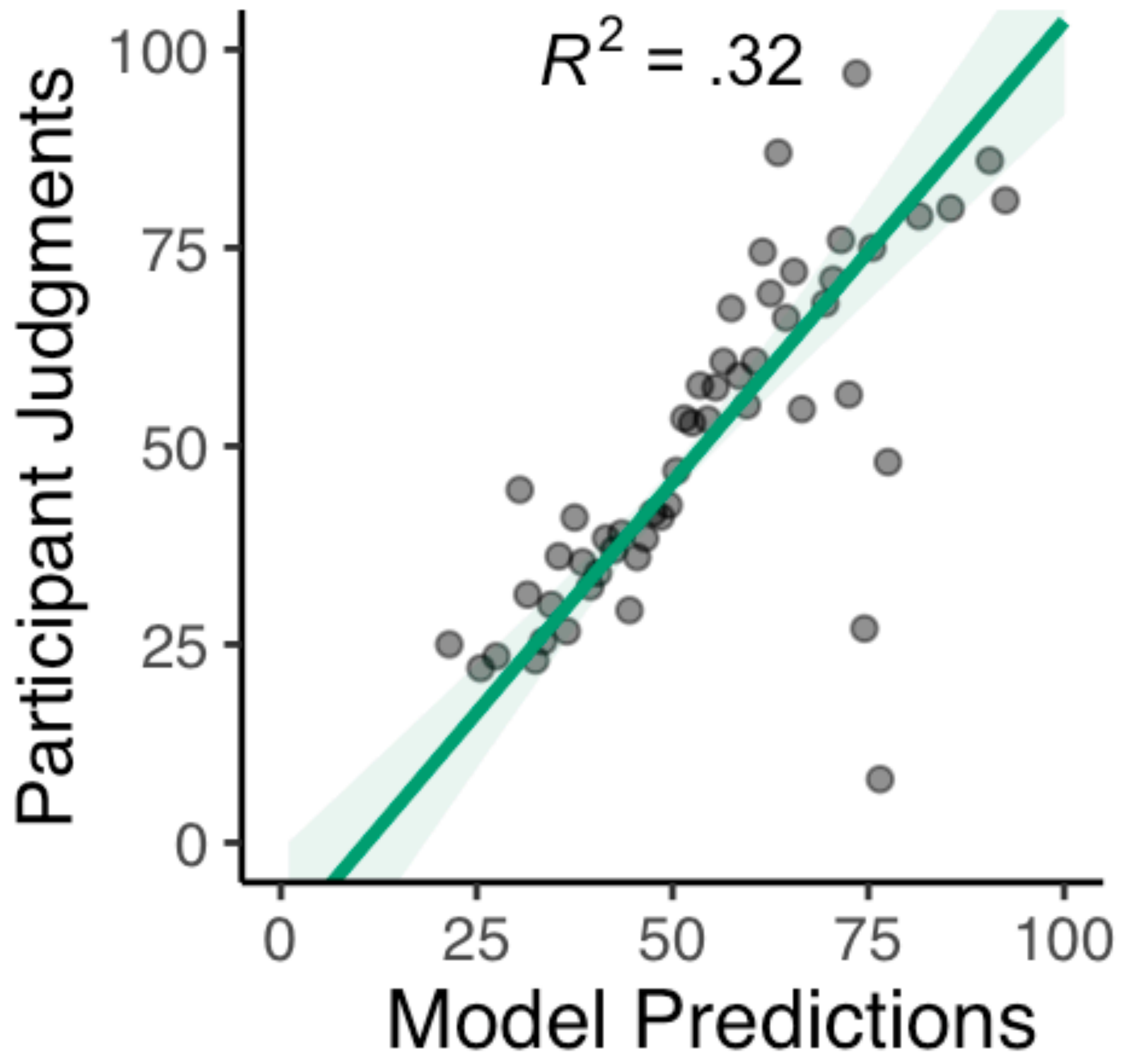




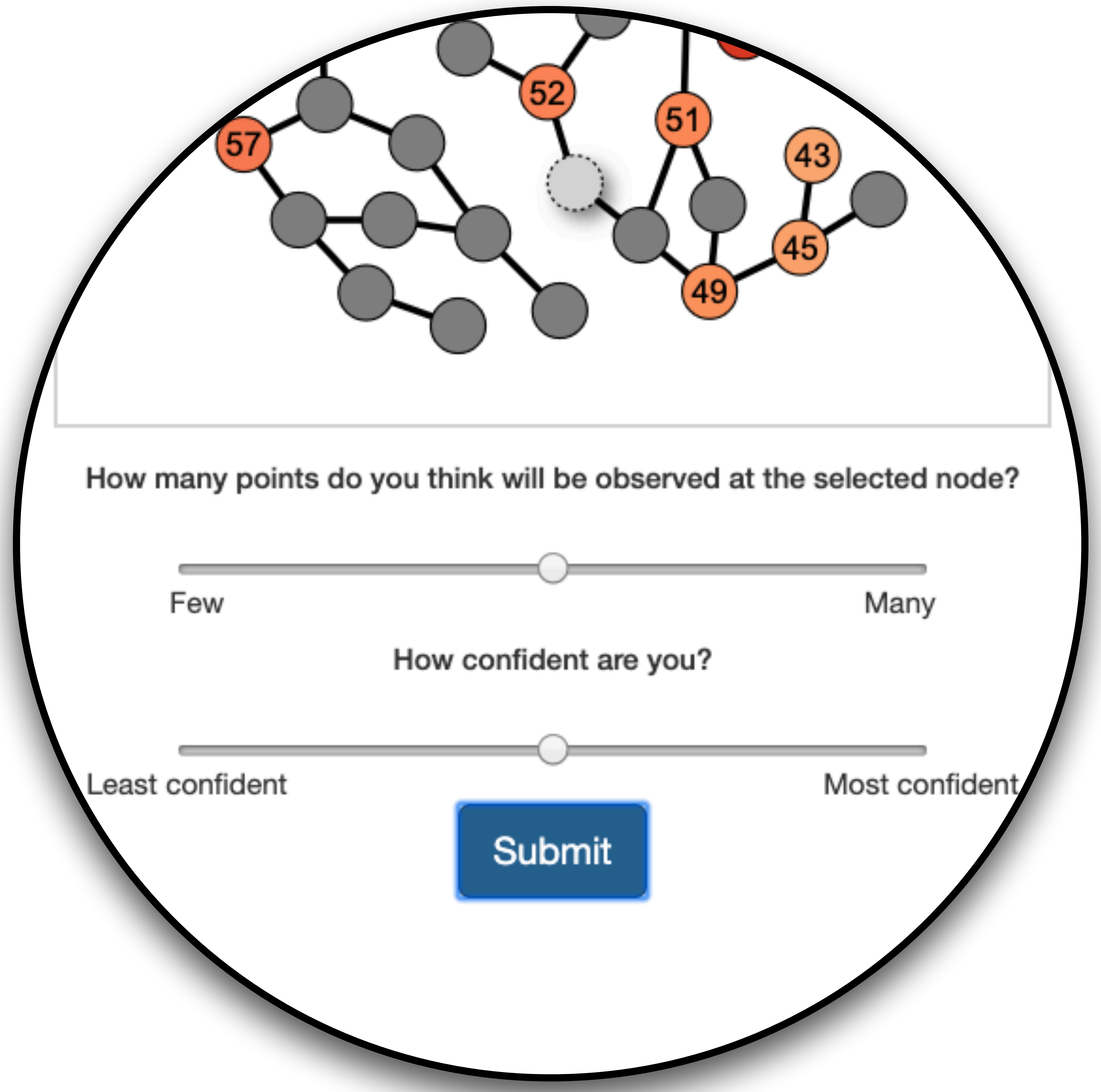
Validation on judgments

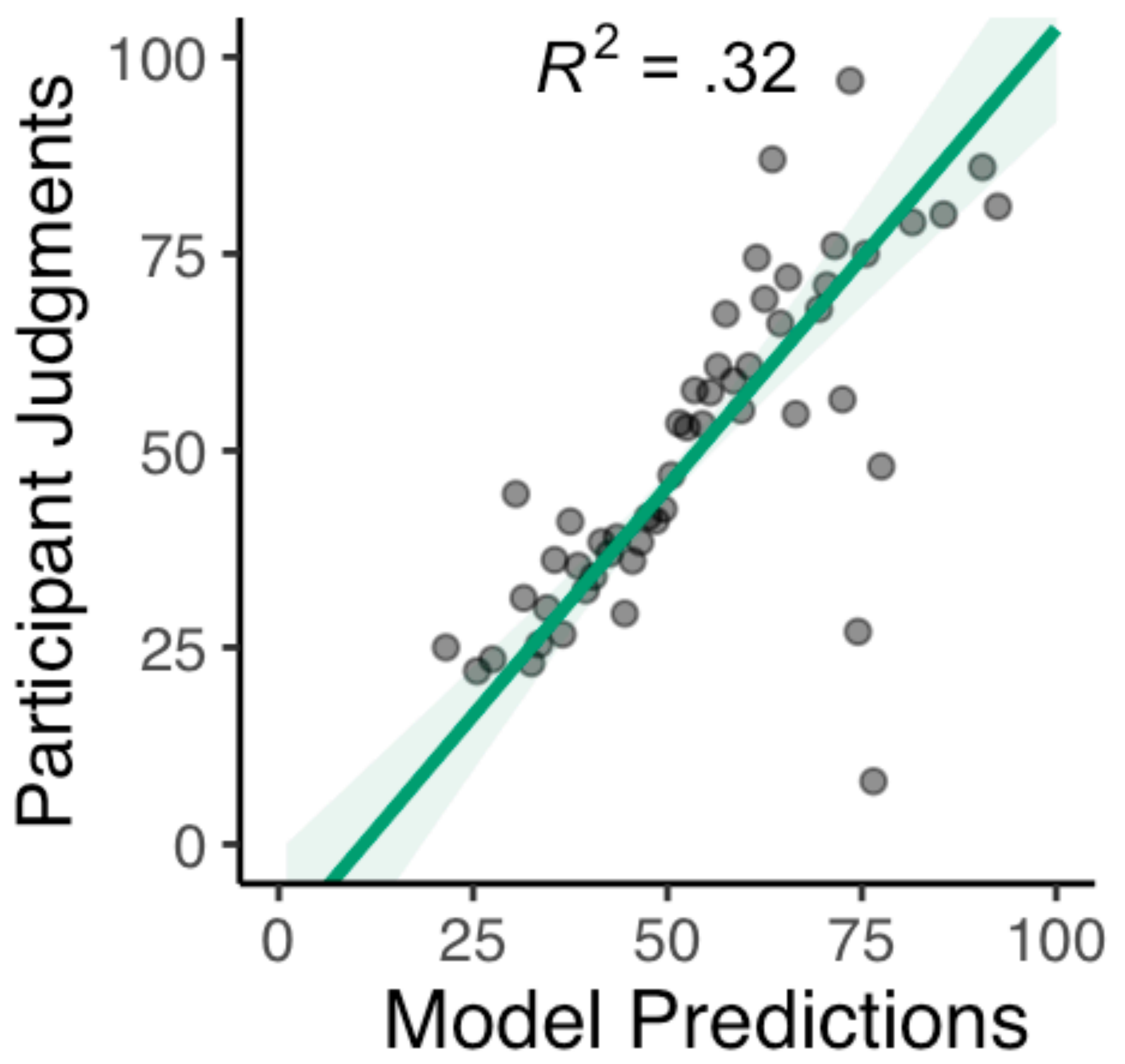
Validation on judgments

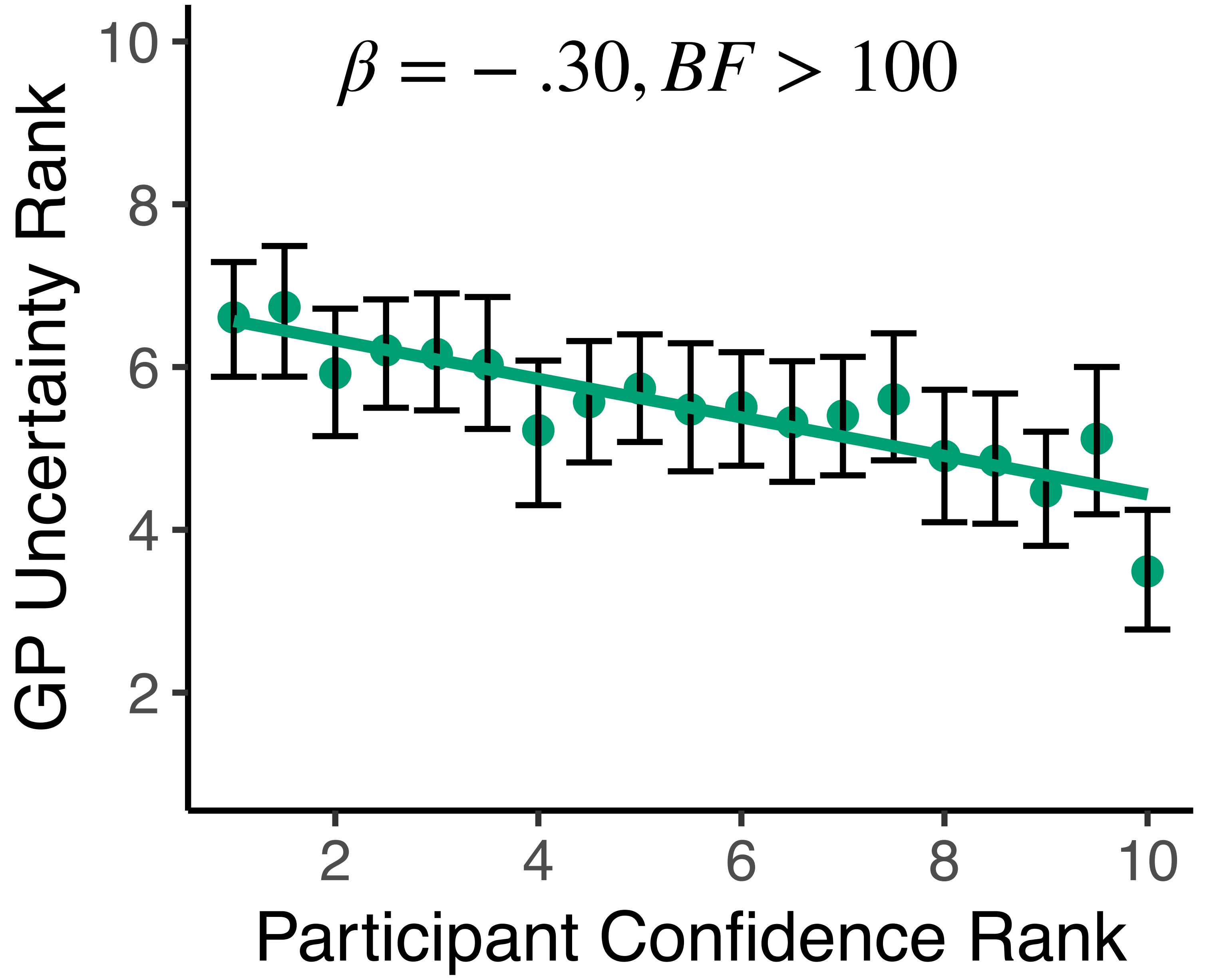




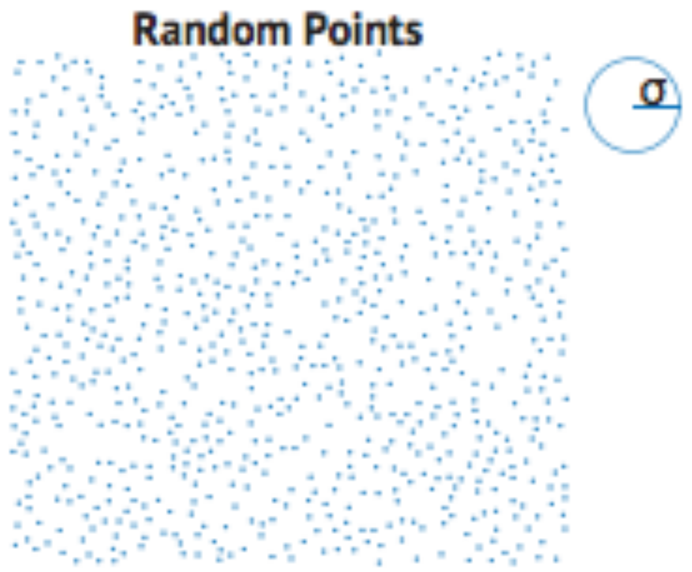
Validation on judgments



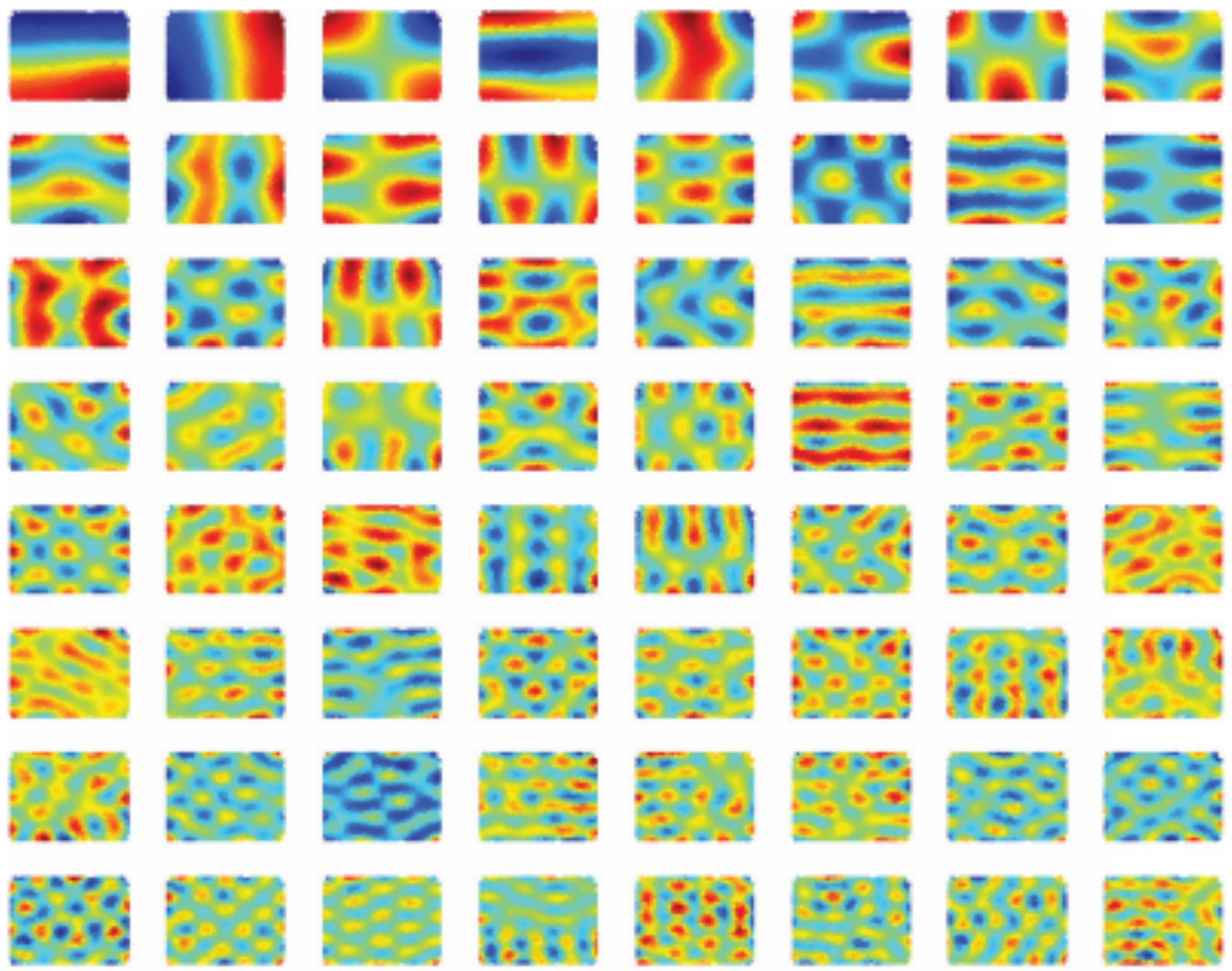


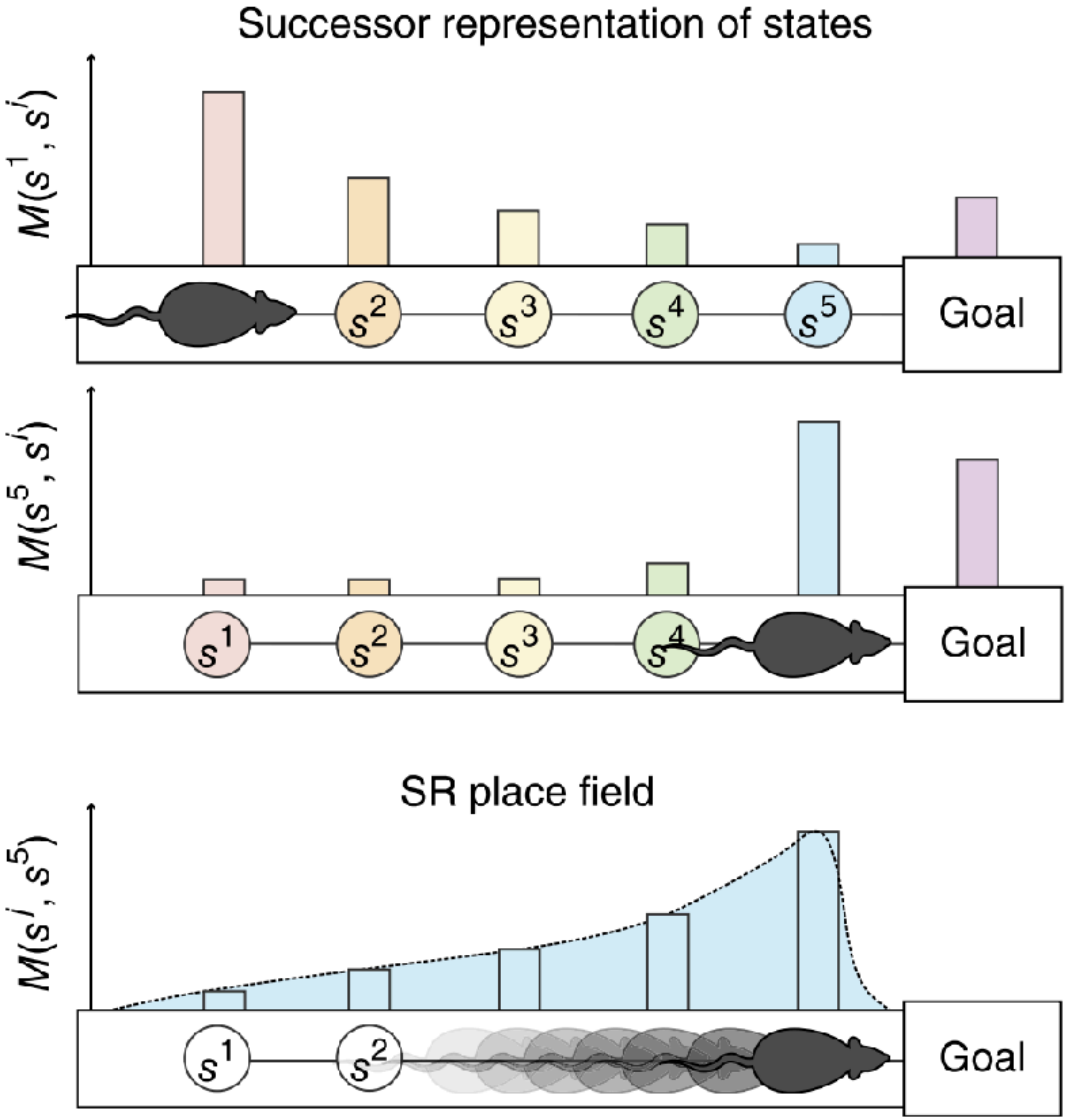


Diffusion Kernel has equivalencies to the Successor Representation



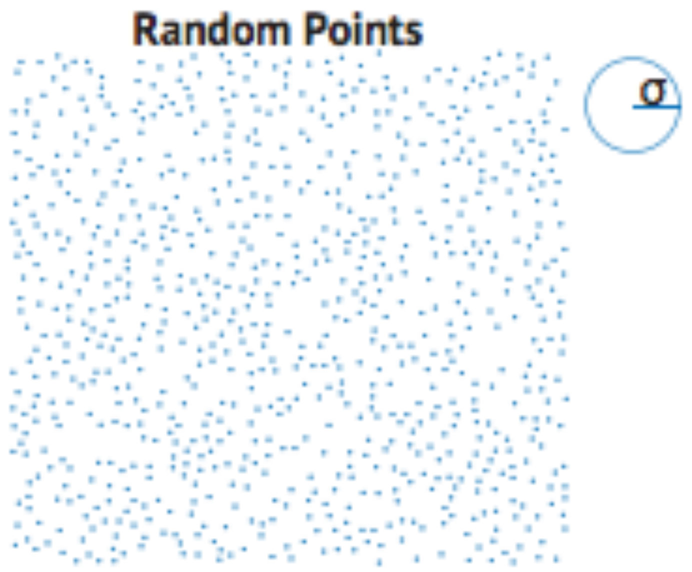
Eigen Vectors



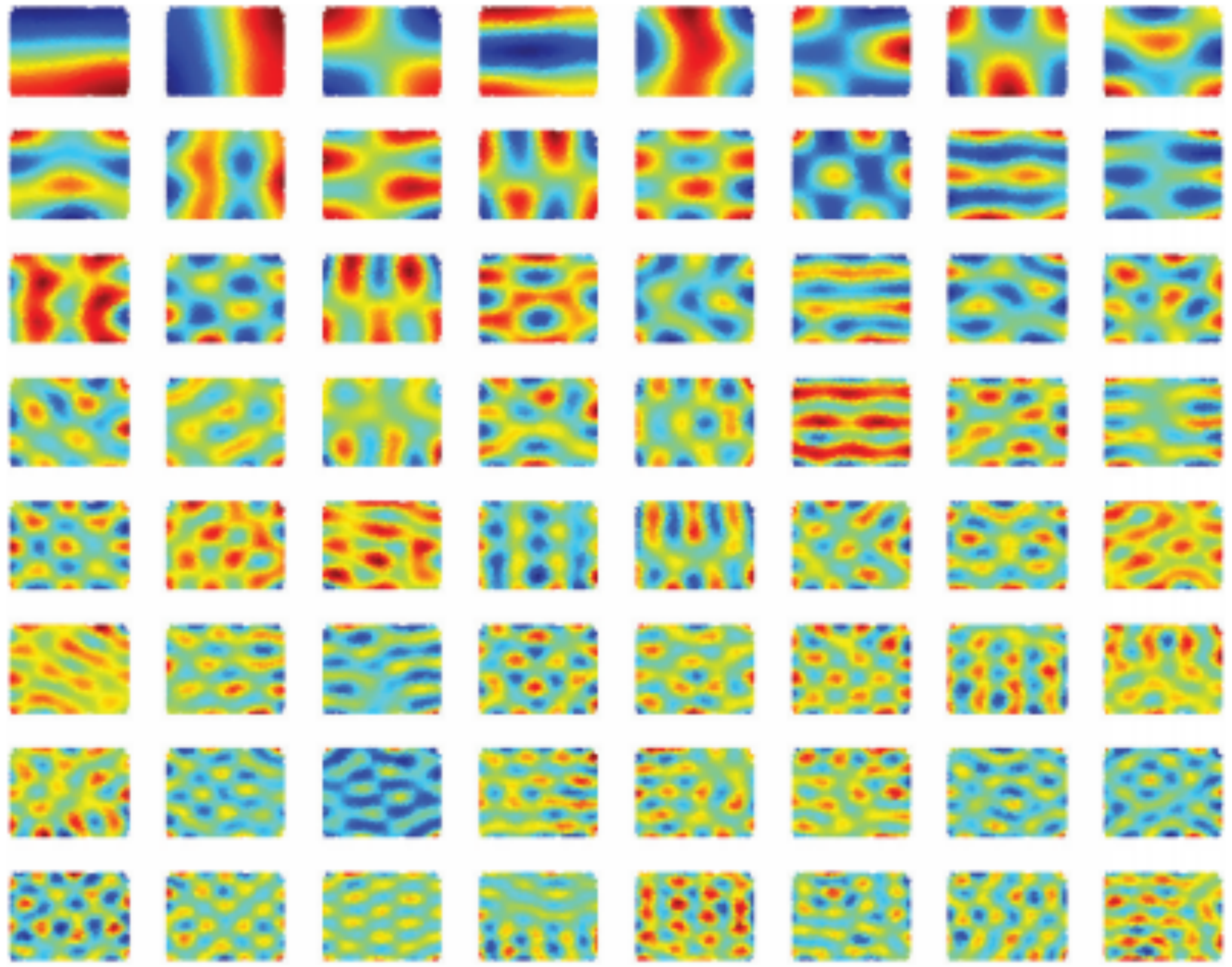


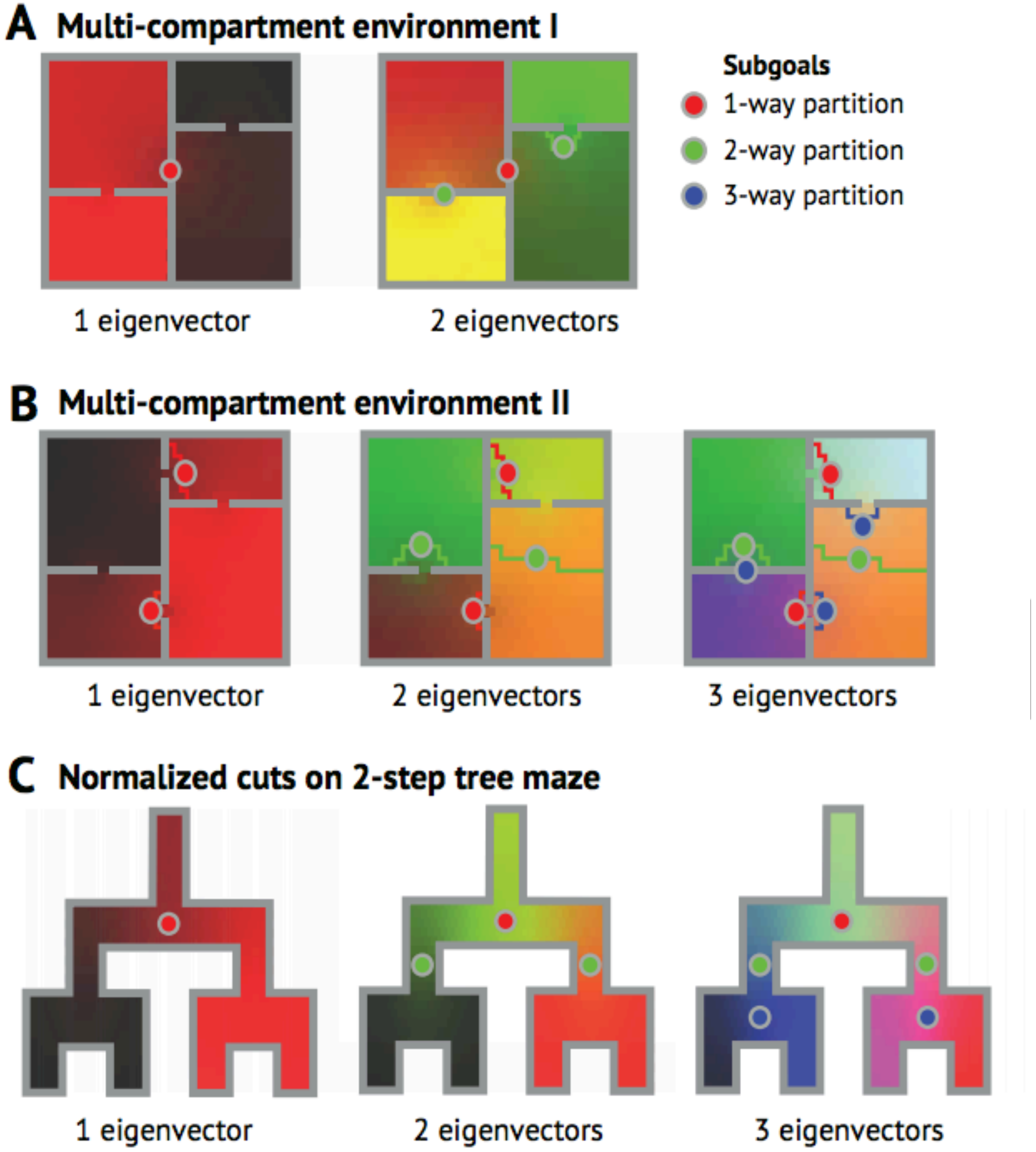
Stachenfeld, Botvinick, & Gershman (*NatNeuro* 2017)

Diffusion Kernel has equivalencies to the Successor Representation



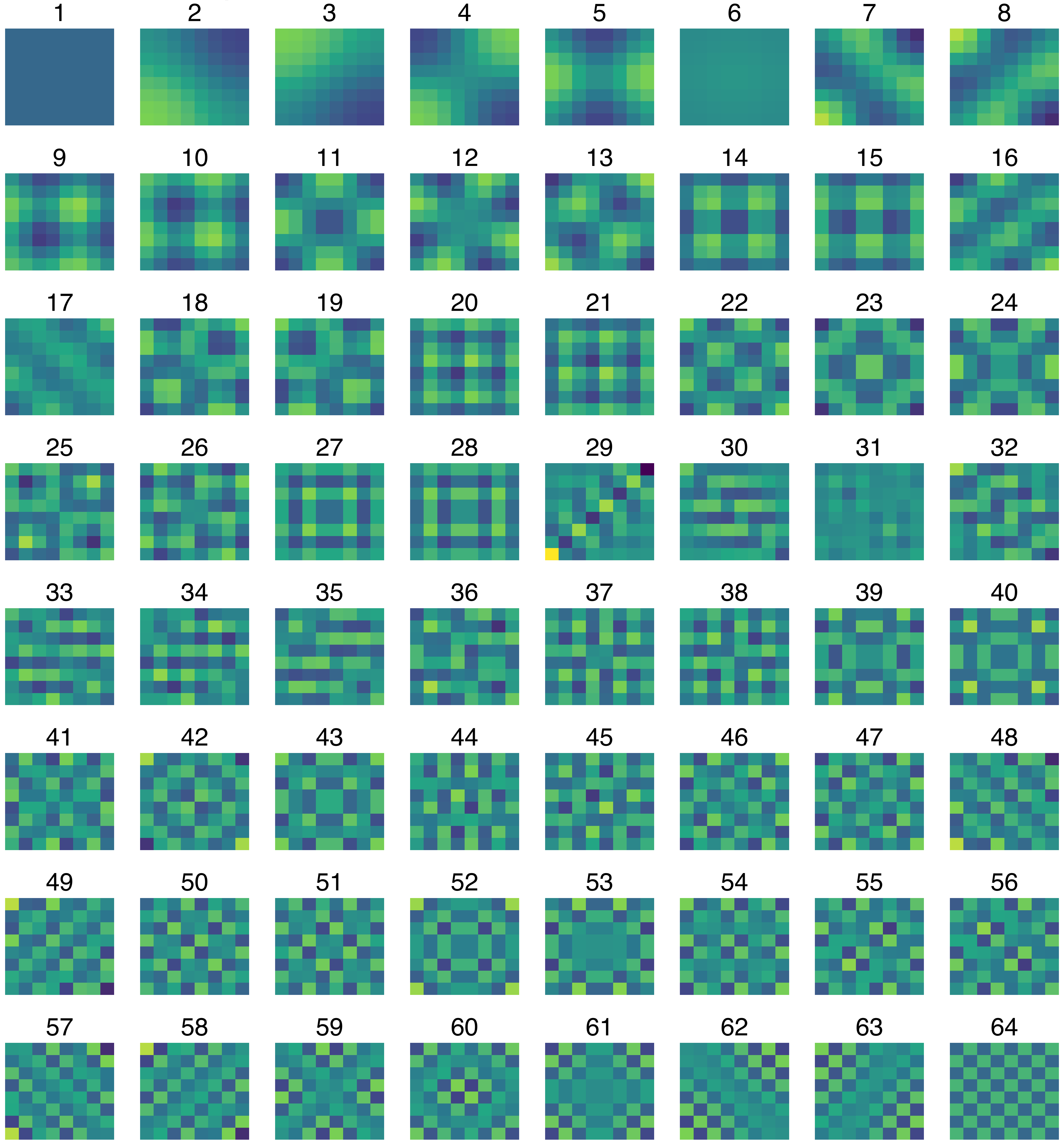
Eigen Vectors

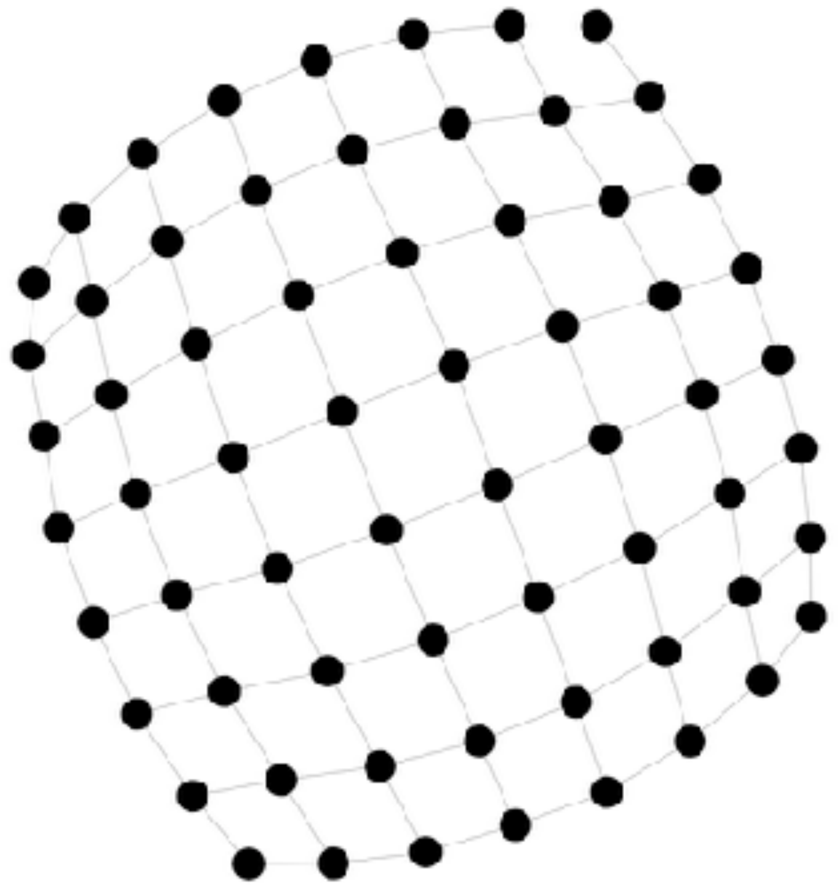




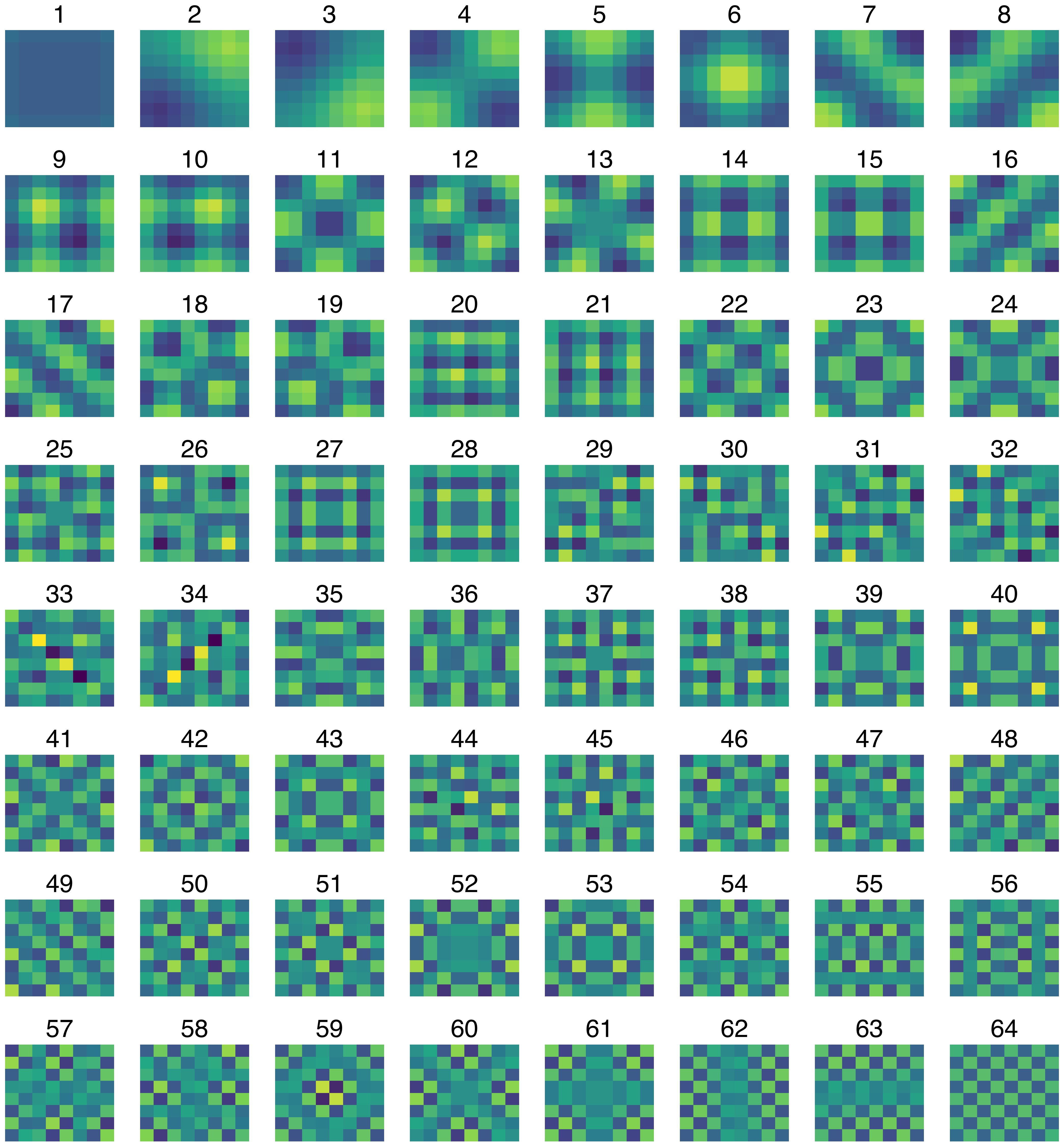
Stachenfeld, Botvinick, & Gershman (*NatNeuro* 2017)

Successor Representation

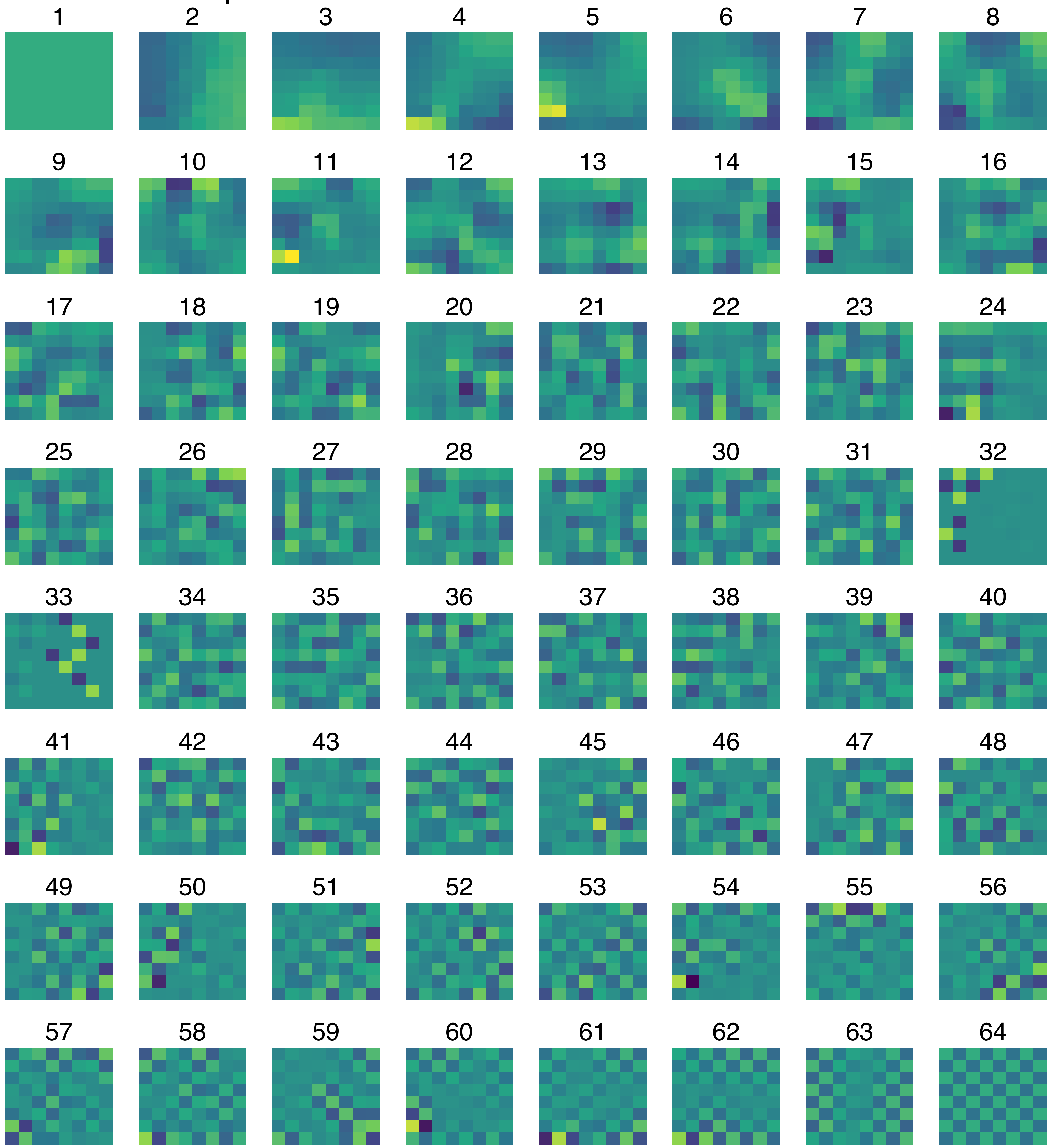


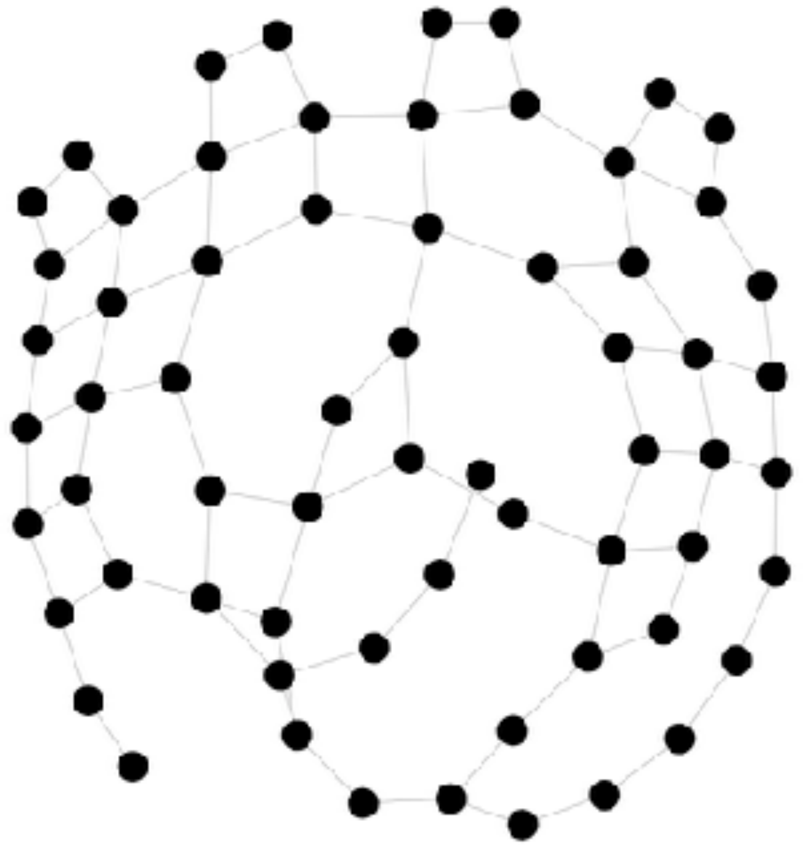


Diffusion Kernel

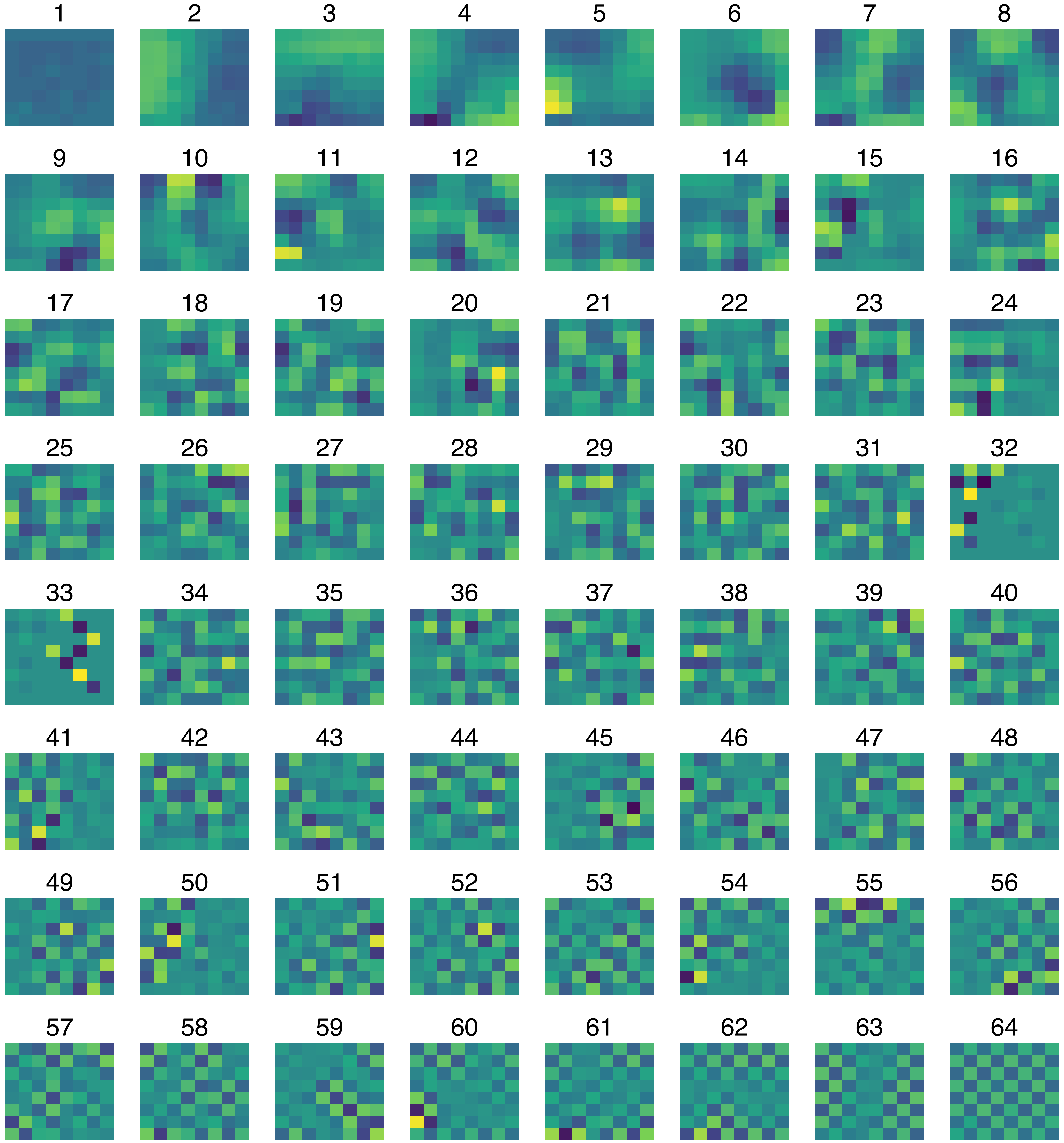


Successor Representation

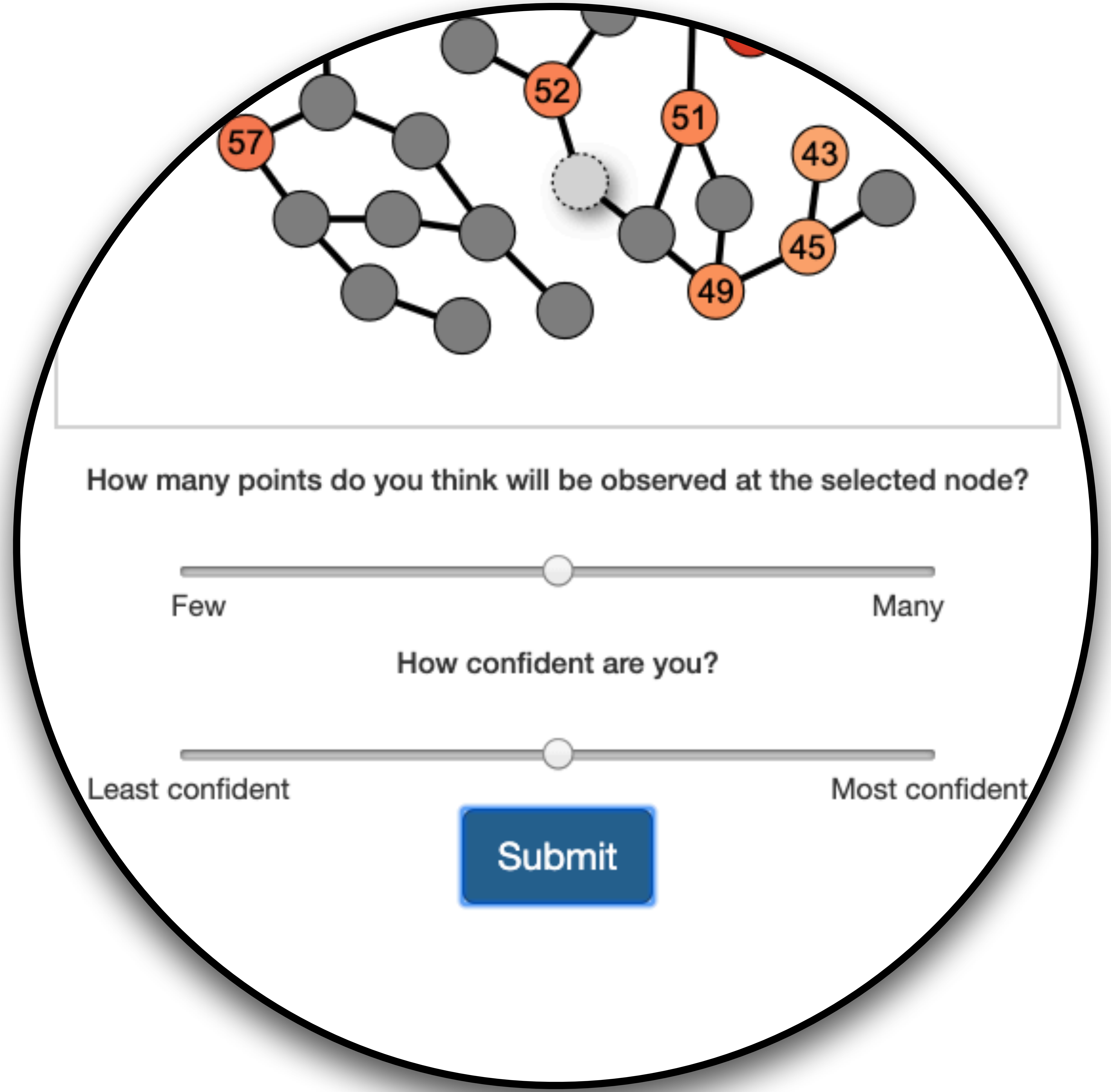




Diffusion Kernel

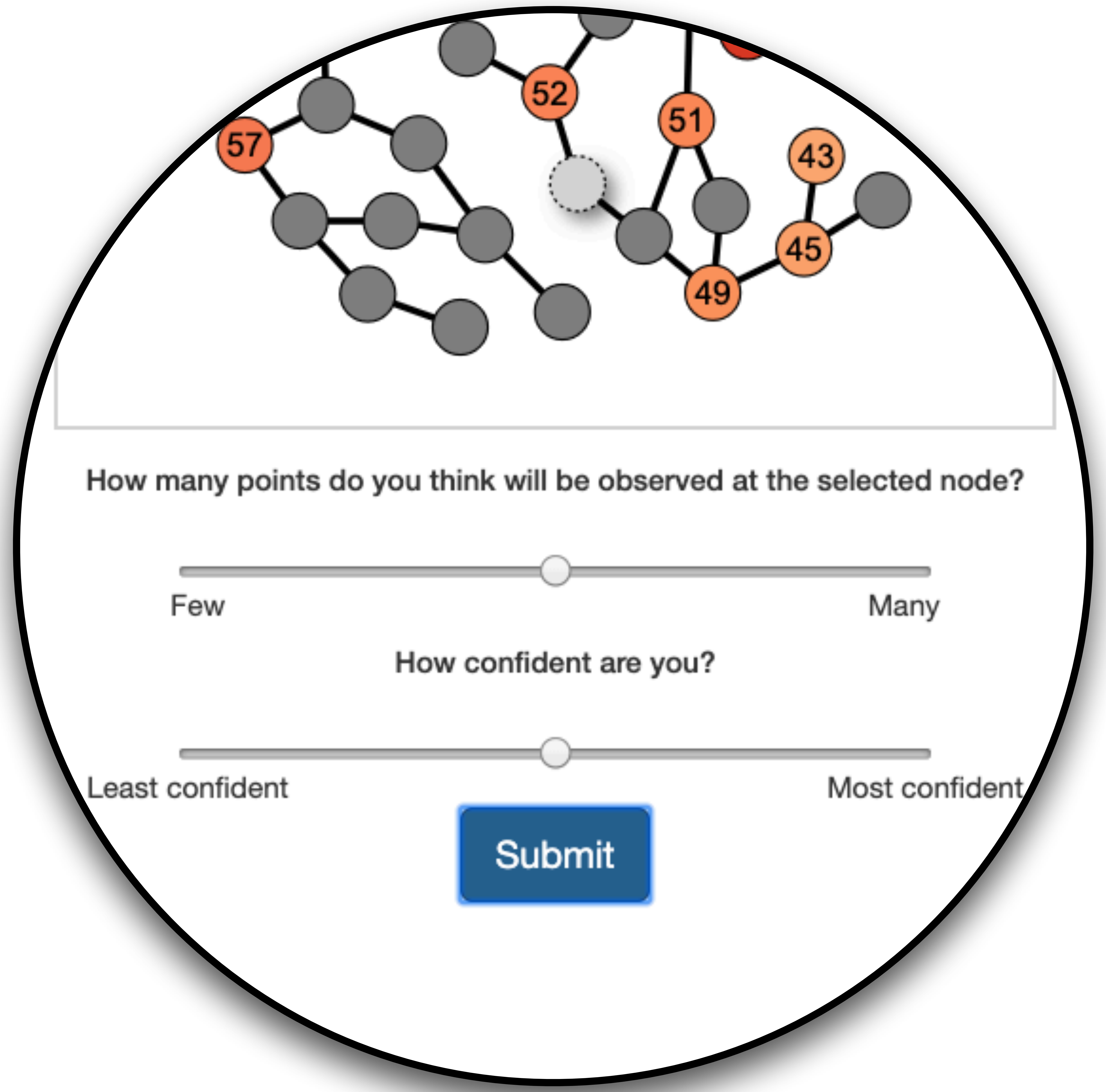


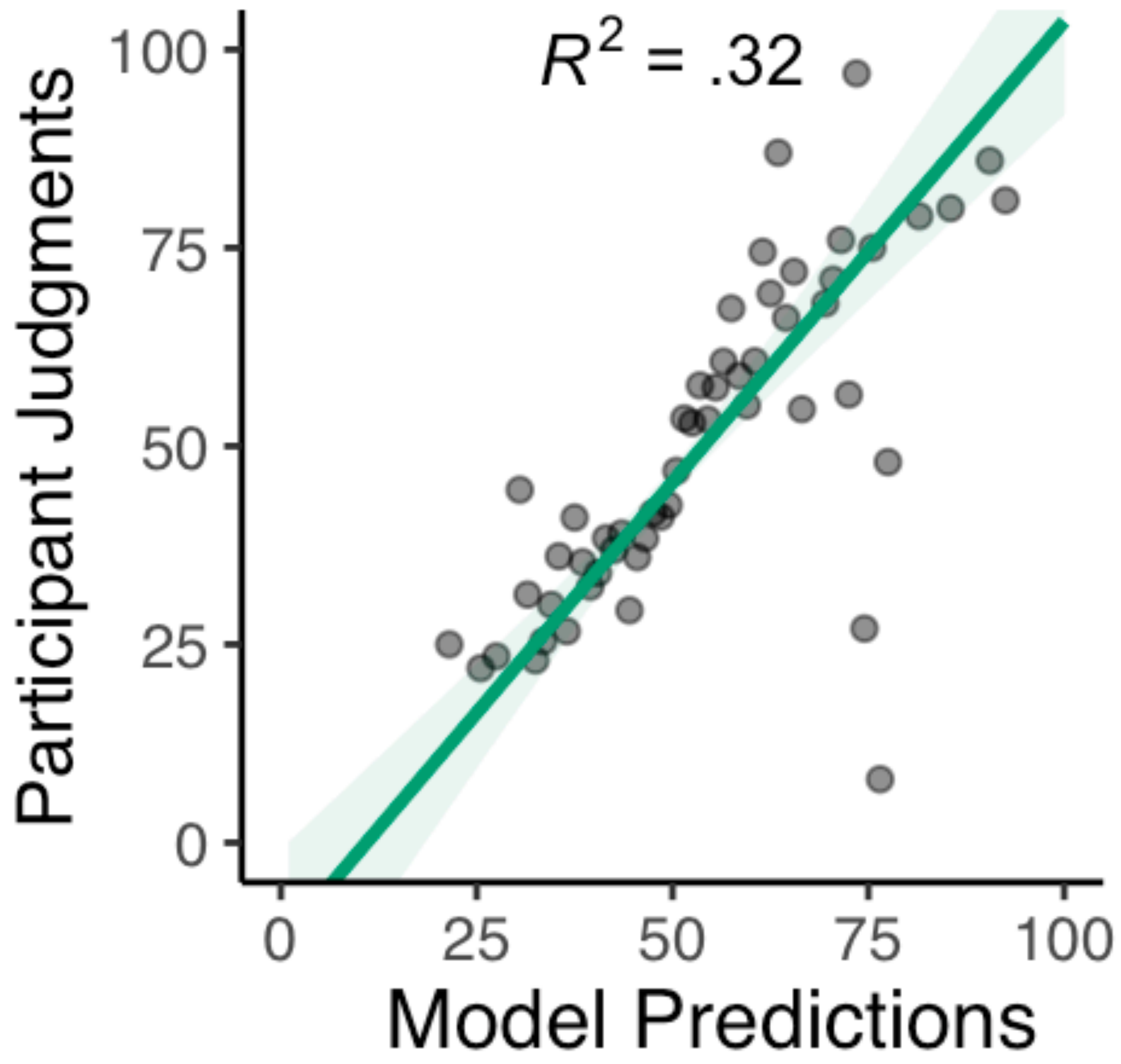
Validation on judgments



Wu, Schulz & Gershman (*CBB* 2021); see also Wu et al,. (*PlosCompBio* 2022)

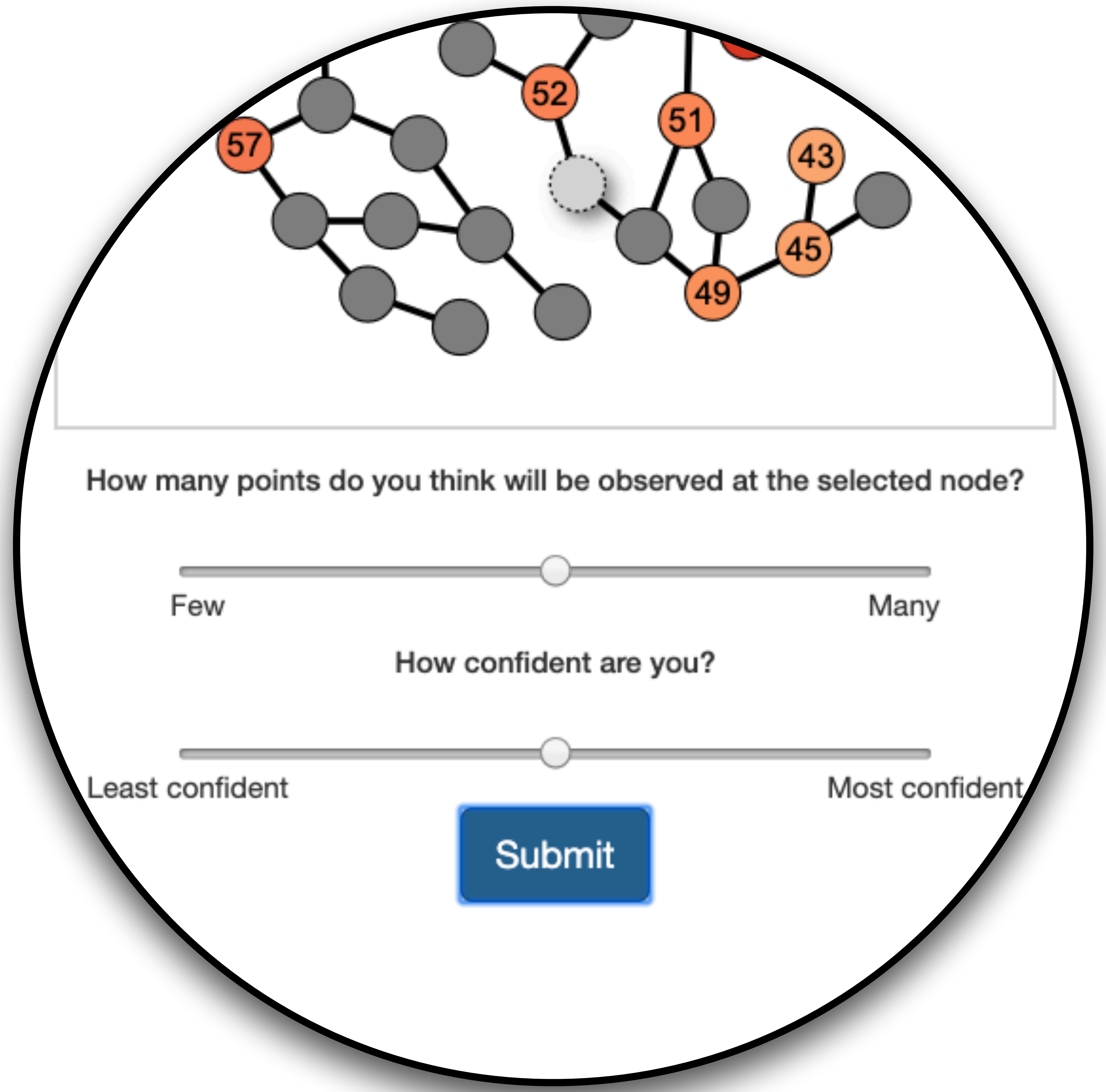
Validation on judgments

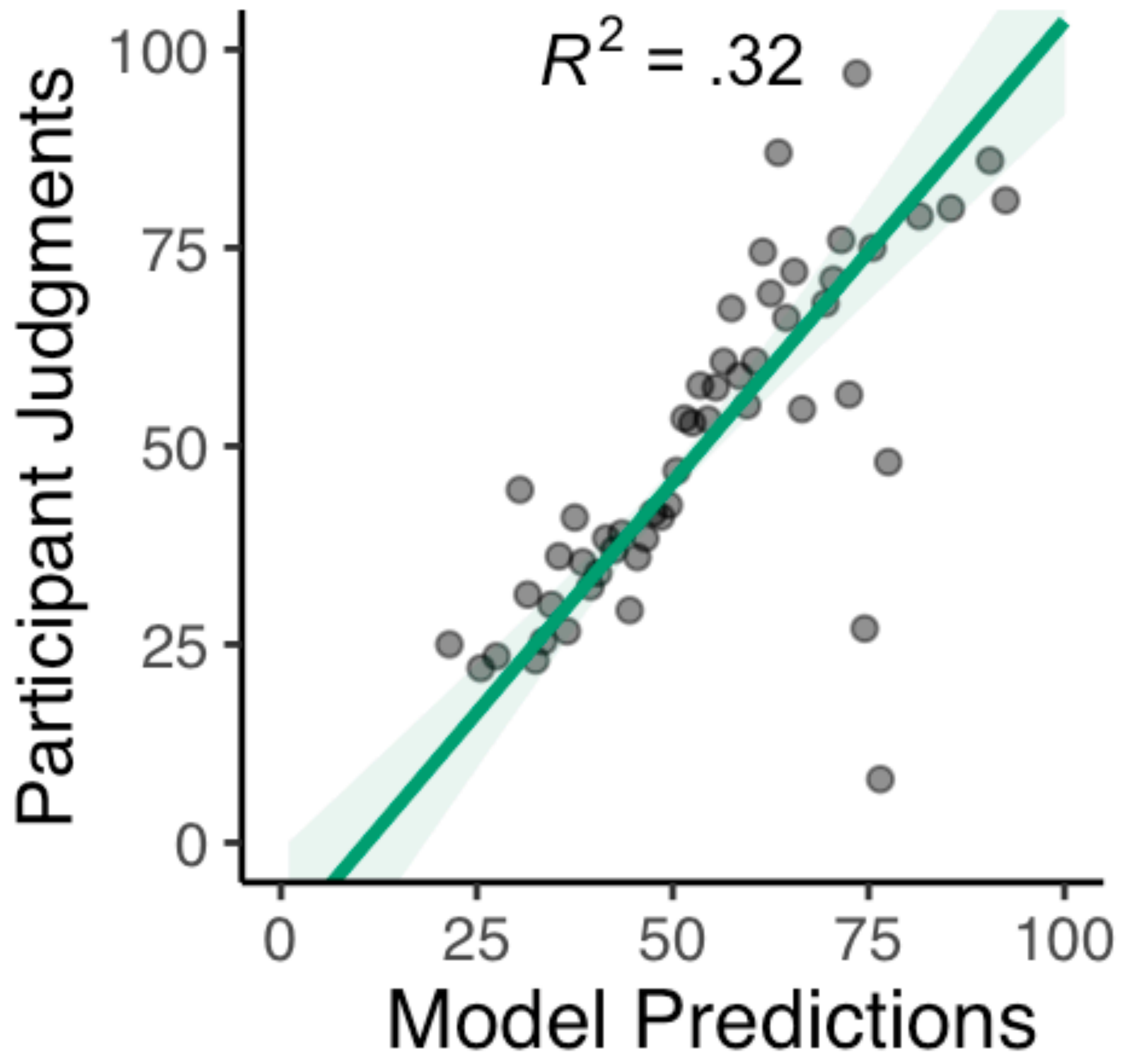


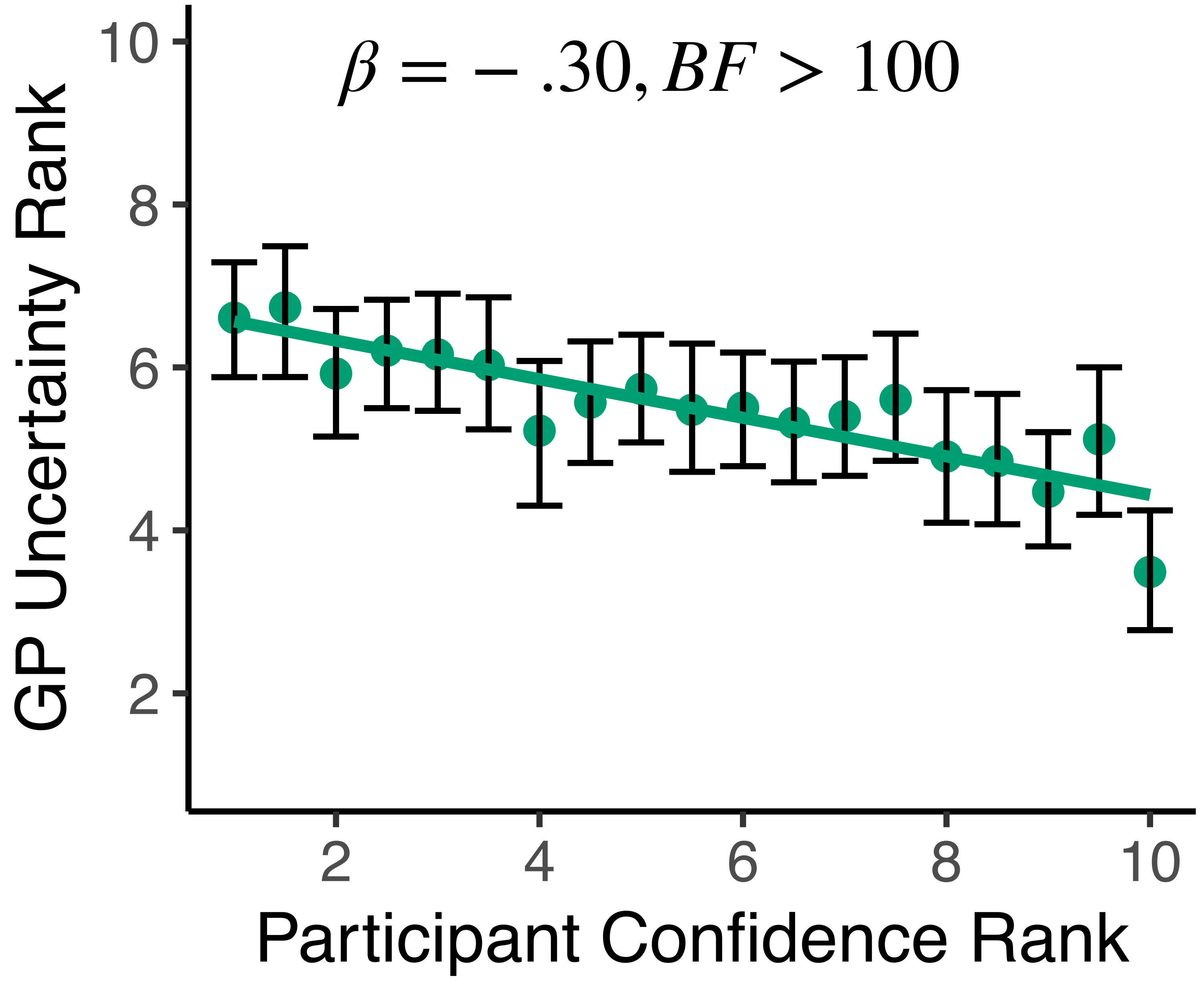


Wu, Schulz & Gershman (*CBB* 2021); see also Wu et al,. (*PlosCompBio* 2022)

Validation on judgments

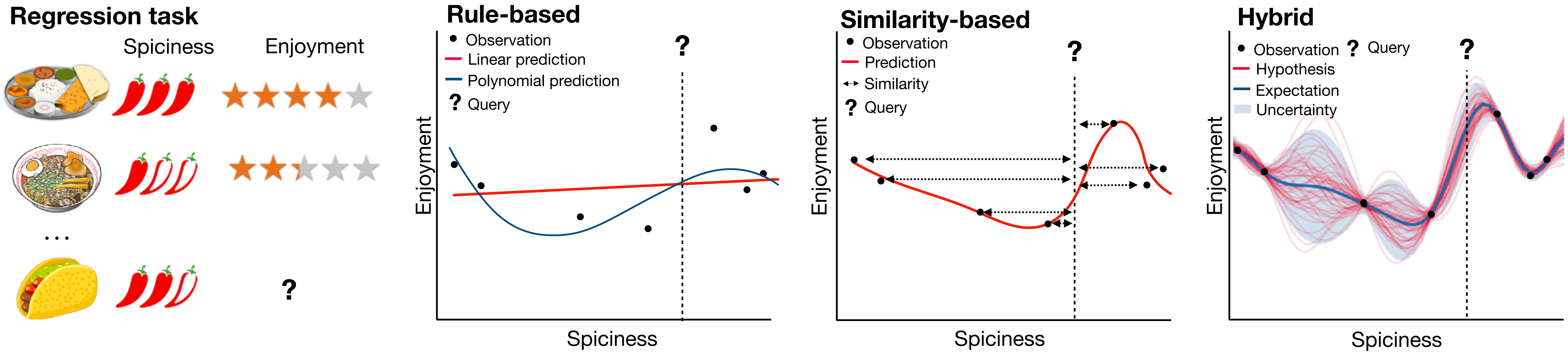






Wu, Schulz & Gershman (*CBB* 2021); see also Wu et al,. (*PlosCompBio* 2022)

Function Learning Summary



- Functions represent candidate hypotheses about the world allowing us to evaluate an infinite range of possibilities through interpolation and extrapolation
- Early **rule-based** approaches lacked flexibility, while **similarity-based** approaches didn't capture human inductive biases
- GP regression is a **hybrid** model, using the principles of Bayesian inference to compute a distribution over candidate hypotheses
- GPs not only capture how humans explicitly learn functions, but also how we implicitly learn a value function to guide our exploration in RL tasks with large search spaces
    - Originally tested in spatial environments (Wu et al,. 2018), but can also be applied to any arbitrary features (Wu et al,. 2020), or even graph-structured environments (Wu et al., 2021)

Next Lecture (in 2 weeks) - Language and Semantics