# Neural network trained with supervision represents uncertainty by nonlinear moments

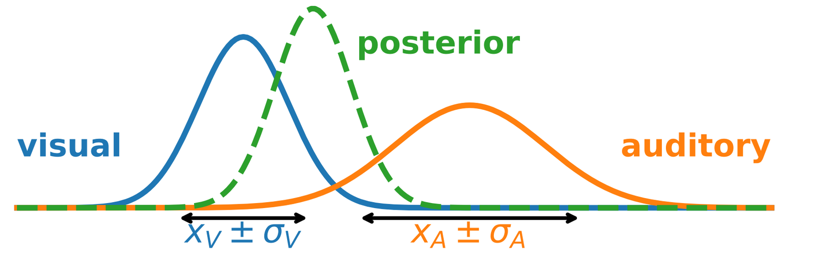
Li Wenliang, Maneesh Sahani

Gatsby Computational Neuroscience Unit, University College London, {kevinli, maneesh}@gatsby.ucl.ac.uk

## Computation with uncertainty

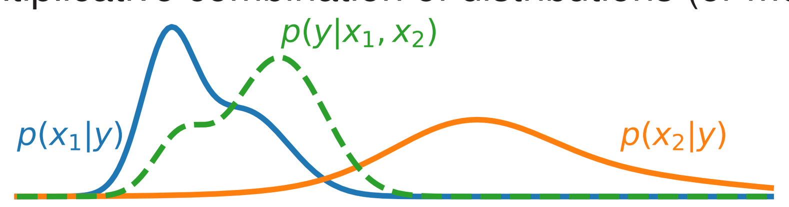
Humans and animals process uncertain sensory inputs reliably as though they are able to manipulate the underlying probability distributions.

- Cue combination (Battaglia et al. 2003)




$$y_{MAP} = \frac{\frac{1}{\sigma_V^2}x_V + \frac{1}{\sigma_a^2}x_A}{\frac{1}{\sigma_V^2} + \frac{1}{\sigma_a^2}}$$

- Causal inference (Körding et al. 2007)
- Risk-based decision making (Whiteley and Sahani 2008)
- Generally, computation with uncertainty requires:
  - multiplicative combination of distributions (or messages)




$$p(y|x_1, x_2) \propto p(x_1|y) \times p(x_2|y)$$

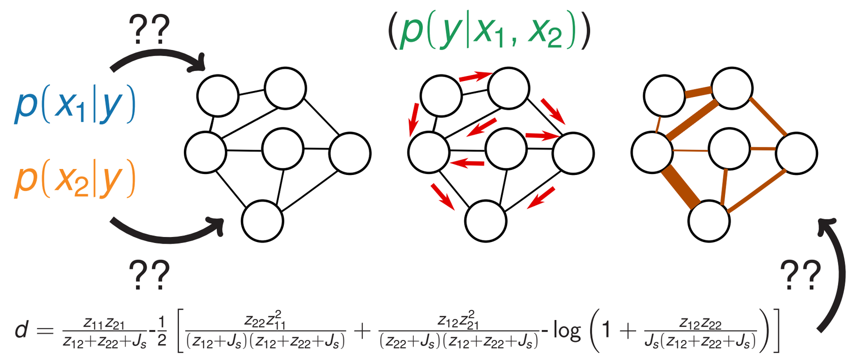
  - marginalization

Four broad hypotheses for how neuronal populations may encode and process uncertainty:

- Linear basis / kernel density estimates (Eliasmith and Anderson 2004)
- (Doubly) Distributional population codes (Zemel et al. 1998, Sahani and Dayan 2003)
- Sampling (Hoyer and Hyvärinen 2003, Orbán et al. 2016)
- Log-linear codes (PPC) (Ma et al. 2006)

**Consistent computation requires a hand-crafted circuit.**

- Supervised networks can learn to interpret noisy inputs, but do not find a consistent representation of uncertainty (Orhan and Ma 2017).
- We seek a neural architecture and learning rule that **automatically acquires consistent representations of and computations with uncertainty** without explicit design.



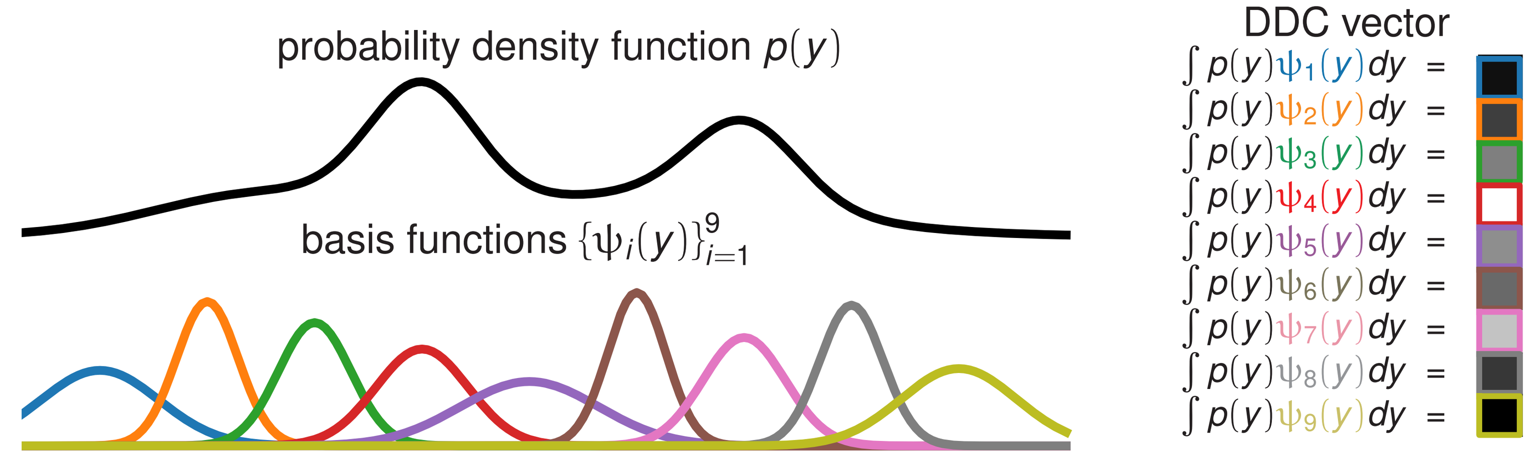

$$d = \frac{z_{11}z_{21}}{z_{12}+z_{22}+J_s} - \frac{1}{2}\left[\frac{z_{22}z_{11}^2}{(z_{12}+J_s)(z_{12}+z_{22}+J_s)} + \frac{z_{12}z_{21}^2}{(z_{22}+J_s)(z_{12}+z_{22}+J_s)} - \log\left(1 + \frac{z_{12}z_{22}}{J_s(z_{12}+z_{22}+J_s)}\right)\right]$$

## References

Battaglia, P. W., Jacobs, R. A., & Aslin, R. N. (2003). Bayesian integration of visual and auditory signals for spatial localization. *Josa a*, *20*(7), 1391–1397.
Eliasmith, C. & Anderson, C. H. (2004). *Neural engineering: Computation, representation, and dynamics in neurobiological systems*. MIT press.
Hoyer, P. O. & Hyvärinen, A. (2003). Interpreting Neural Response Variability as Monte Carlo Sampling of the Posterior. In S. Becker, S. Thrun, & K. Obermayer (Eds.), *Nips* (pp. 293–300).
Körding, K. P., Beierholm, U., Ma, W. J., Quartz, S., Tenenbaum, J. B., & Shams, L. (2007). Causal Inference in Multisensory Perception. *PLOS ONE*, *2*(9), 1–10.
Ma, W. J., Beck, J. M., Latham, P. E., & Pouget, A. (2006). Bayesian inference with probabilistic population codes. *Nature Neuroscience*, *9*(11), 1432–1438.
Orbán, G., Berkes, P., Fiser, J., & Lengyel, M. (2016). Neural Variability and Sampling-Based Probabilistic Representations in the Visual Cortex. *Neuron*, *92*(2), 530–543.
Orhan, A. E. & Ma, W. J. (2017). Efficient probabilistic inference in generic neural networks trained with non-probabilistic feedback. *Nature Communications*, *8*(1), 138.
Sahani, M. & Dayan, P. (2003). Doubly distributional population codes: simultaneous representation of uncertainty and multiplicity. *Neural Computation*, *15*(10), 2255–2279.
Vértes, E. & Sahani, M. (2018). Flexible and accurate wake-sleep learning in deep exponential family models (submitted). *AISTATS*.
Whiteley, L. & Sahani, M. (2008). Implicit knowledge of visual uncertainty guides decisions with asymmetric outcomes. *Journal of Vision*, *8*(3), 2.
Zemel, R. S., Dayan, P., & Pouget, A. (1998). Probabilistic Interpretation of Population Codes. *Neural Computation*, *10*(2), 403–430.

## The distributed distributional code (DDC)

- Belief about $\mathbf{y}$ represented by linear projections of density $p(\mathbf{y})$ on basis functions $\psi_i$:




(cf. distributional population codes Zemel et al. 1998; Sahani and Dayan 2003)

### *Representation*

- Expectations (or moments) define an **exponential family** of beliefs by maximum entropy.
- E.g. Gaussians are defined by linear and quadratic functions (first and second moments):

$$[r_1, r_2] = \left[\mathbb{E}_{p(y)}[\psi_1(y)], \mathbb{E}_{p(y)}[\psi_2(y)]\right]$$

where $\psi_1(y) = y$ and $\psi_2(y) = y^2$.

- Arbitrary functions define more complicated families:

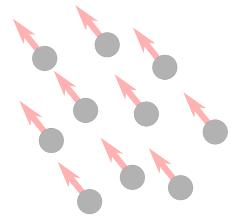
$$\mathbf{r} = \mathbb{E}_{p(y)}[\psi_1(y), \cdots, \psi_{n-1}(y), \psi_n(y)] \Rightarrow p(\mathbf{y}|\boldsymbol{\theta}) \propto \exp(\boldsymbol{\theta} \cdot \boldsymbol{\psi}(\mathbf{y})) \text{ such that } \mathbb{E}_{p(\mathbf{y}|\boldsymbol{\theta})}[\boldsymbol{\psi}(\mathbf{y})] = \mathbf{r}$$

The rates $\mathbf{r}$ are the *mean parameters* of the distribution.

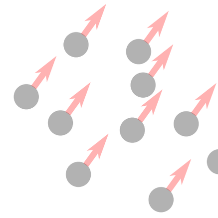
- Can encode multiplicity and uncertainty:

*uncertainty: left or right*

| $v_L$ | $v_R$ | $P(v_L, v_R)$ |
|---|---|---|
| 1.0 | 0.0 | 0.5 |
| 0.0 | 1.0 | 0.5 |
| 0.5 | 0.5 | 0.0 |



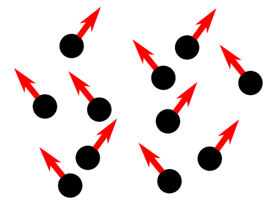
or



$$\mathbb{E}_{p(v_L, v_R)}[\psi(v_L, v_R)] = 0.5\psi(1.0, 0.0) + 0.5\psi(0.0, 1.0)$$

*multiplicity: mixed left and right*

| $v_L$ | $v_R$ | $P(v_L, v_R)$ |
|---|---|---|
| 1.0 | 0.0 | 0.0 |
| 0.0 | 1.0 | 0.0 |
| 0.5 | 0.5 | 1.0 |



$$\mathbb{E}_{p(v_L, v_R)}[\psi(v_L, v_R)] = \psi(0.5, 0.5)$$

### *Computation*

- Easy to approximate expectations of functions given a rich set of basis functions:

$$g(\mathbf{y}) \approx \sum_i \alpha_i \psi_i(\mathbf{y}) \Longrightarrow \mathbb{E}_{p(\mathbf{y})}[g(\mathbf{y})] \approx \mathbb{E}_{p(\mathbf{y})}\left[\sum_i \alpha_i \psi_i(\mathbf{y})\right] = \sum_i \alpha_i \mathbb{E}_{p(\mathbf{y})}[\psi_i(\mathbf{y})] = \sum_i \alpha_i r_i$$

- Can implement conditioning and message passing:

$$p(y|x) \propto p(y)p(x|y) \Leftrightarrow \mathbb{E}_{p(y|x)}[\psi(y)] = \mathcal{W}\phi(x)$$

### *Learning*

- Each neuron must estimate an expectation: easy to do with supervised learning.
- In fully observed models, probabilistic computations can be learned from training data:

$$\mathcal{W} = \mathrm{Cov}[\psi(y), \phi(x)]\mathrm{Cov}[\phi(x), \phi(x)]^{-1}$$

- State-of-the-art performance using wake-sleep (Vértes and Sahani 2018 — **Poster II-57**).

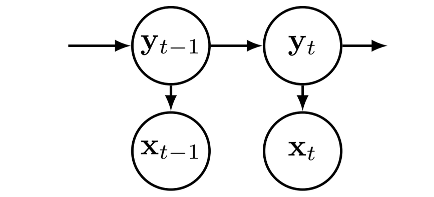
## Key question

**Can a network trained *without explicit supervision of latents* learn**

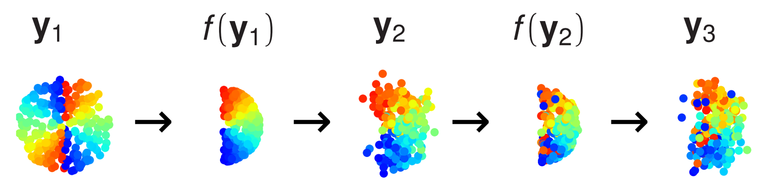
- **to represent probabilistic beliefs?**
- **to propagate uncertainty (message passing)?**

## A task that requires uncertainty computation

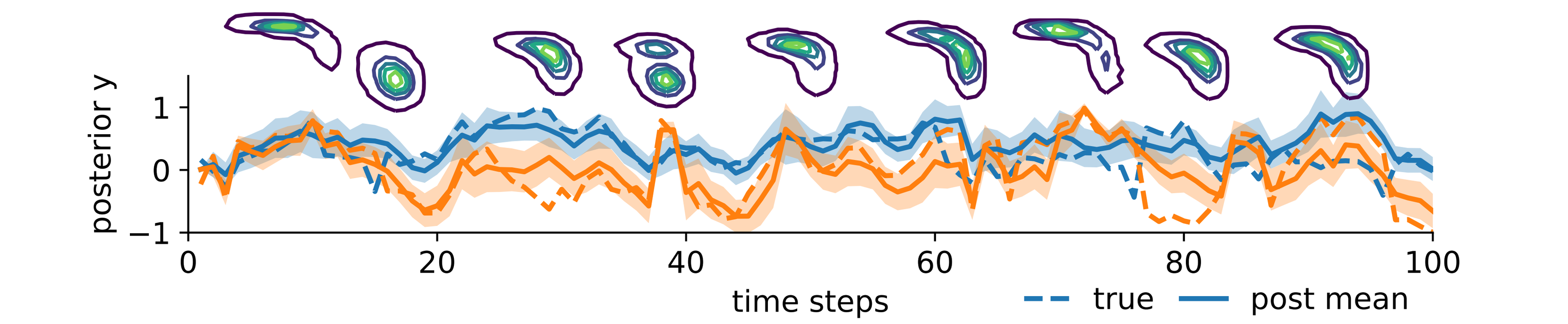
**Data**: state-space model




$\mathbf{y}_t \sim \mathrm{Normal}(f(\mathbf{y}_{t-1}), \Sigma)$
$\mathbf{x}_t \sim \mathrm{Poisson}(\mathbf{G}e^{\mathbf{B}\mathbf{y}_t})$




**Task**: predict $\mathbf{y}_t$ given $\mathbf{x}_{1:t}$ so that the MSE $\langle\|\mathbf{y}_t - \hat{\mathbf{y}}_t\|^2\rangle$ is minimized.
**Example**: tracking a prey moving behind bushes in real time.
**Answer**: $\mathbb{E}_{p(\mathbf{y}_t|\mathbf{x}_{1:t})}[\mathbf{y}_t]$, requires good representation of $p(\mathbf{y}_t|\mathbf{x}_{1:t})$.
**Constraint**: the brain does not have access to the full history $\mathbf{x}_{1:t}$ but only has a representation of it in memory, necessitating effective encoding of posterior.




## DDC update of latent belief state: bilinear message passing

$$\underbrace{p(\mathbf{y}_t|\mathbf{x}_{1:t})}_{\text{new belief}} \propto \int p(\mathbf{y}_{t-1}|\mathbf{x}_{1:t-1})p(\mathbf{y}_t|\mathbf{y}_{t-1})p(\mathbf{x}_t|\mathbf{y}_t)d\mathbf{y}_{t-1}$$

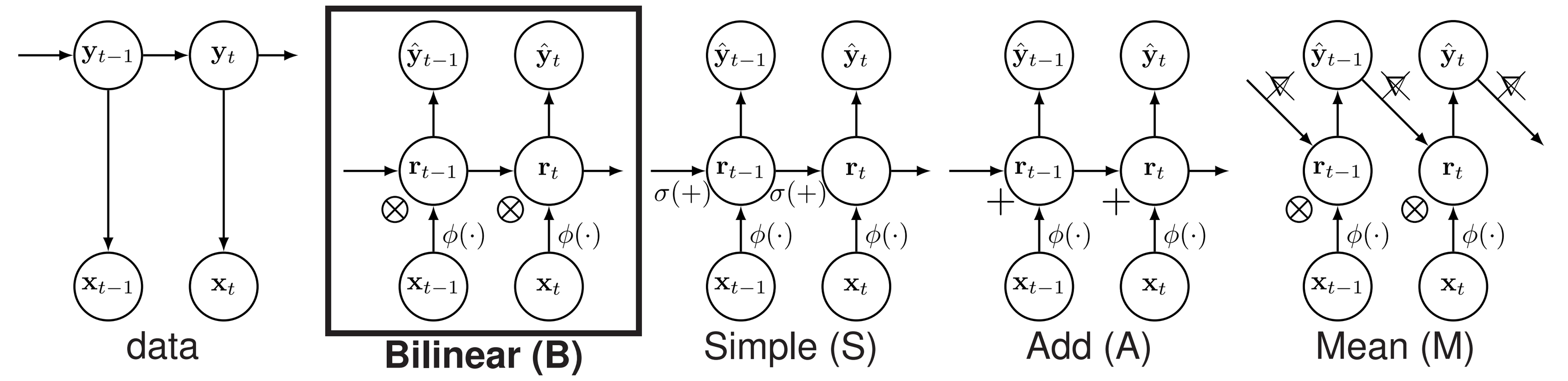
$$= \int p(\mathbf{y}_{t-1}|\mathbf{x}_{1:t-1})p(\mathbf{y}_t|\mathbf{y}_{t-1}) \int p(\mathbf{X}_t|\mathbf{y}_t)\delta(\mathbf{X}_t - \mathbf{x}_t)d\mathbf{X}_t d\mathbf{y}_{t-1}$$

$$= \int\int \underbrace{p(\mathbf{y}_{t-1}|\mathbf{x}_{1:t-1})}_{\text{memory}} \underbrace{p(\mathbf{y}_t|\mathbf{y}_{t-1})p(\mathbf{X}_t|\mathbf{y}_t)}_{\text{model}} \underbrace{\delta(\mathbf{X}_t - \mathbf{x}_t)}_{\text{feature}} d\mathbf{X}_t d\mathbf{y}_{t-1}$$

New belief is a **bilinear** function of memory $\mathbf{r}_{t-1}$ and observation feature $\phi_t$. If we represent $p(\mathbf{y}_t|\mathbf{x}_{1:t})$ by a vector $\mathbf{r}_t$, then

$$r_t^{(i)} = \mathbf{r}_{t-1} \cdot \mathbf{W}^{(i)} \cdot \phi(\mathbf{x}_t)$$
$$\mathbf{r}_t = \mathbf{W} \cdot \mathrm{vec}(\mathbf{r}_{t-1} \otimes \phi_t)$$

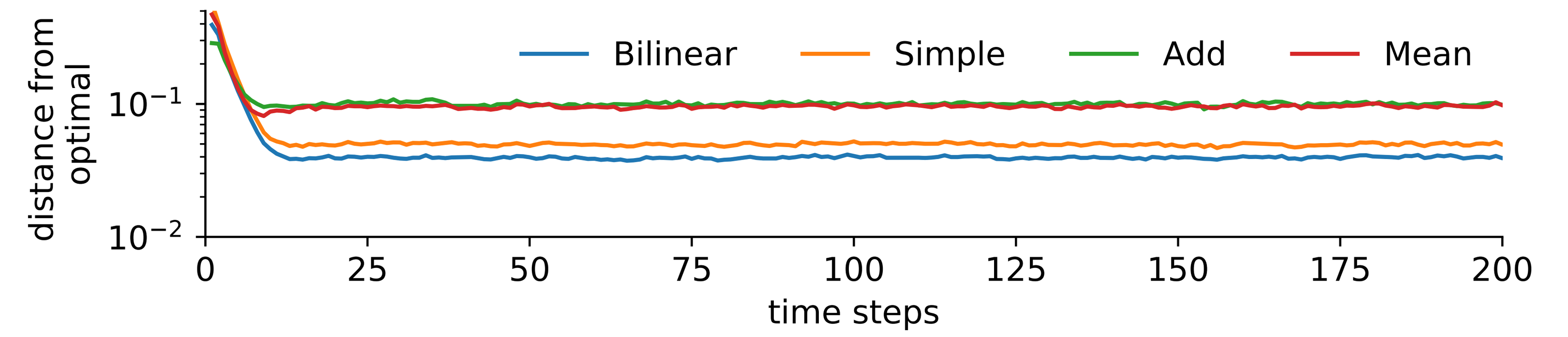
## DDC-consistent recurrent neural network




- Transform $\mathbf{x}_t$ into a DDC vector: $\phi_t = \tanh(\mathbf{V}\mathbf{x}_t) = \mathbb{E}_{\delta(\mathbf{X}_t - \mathbf{x}_t)}[\tanh(\mathbf{V}\mathbf{X}_t)]$.
- B: $\mathbf{r}_t = \mathbf{W} \cdot \mathrm{vec}(\mathbf{r}_{t-1} \otimes \phi_t)$  S: $\mathbf{r}_t = \tanh(\mathbf{W}_1\mathbf{r}_{t-1} + \mathbf{W}_2\phi_t)$  A: $\mathbf{r}_t = \tanh(\mathbf{W}_1\mathbf{r}_{t-1}) + \tanh(\mathbf{W}_2\phi_t)$
- M: follows bilinear update but only passes on the previous predicted mean.
- Training: $\hat{\mathbf{y}}_t = \mathbf{U}\mathbf{r}_t$, minimize $\langle\|\mathbf{y}_t - \hat{\mathbf{y}}_t\|^2\rangle$ over $\mathbf{U}$, $\mathbf{V}$ and $\mathbf{W}$ by back-propagation through time.
- **Importantly, the neurons are not enforced to have a DDC representation.**

## Results

- The Bilinear network makes predictions closer to the particle filter than alternatives.



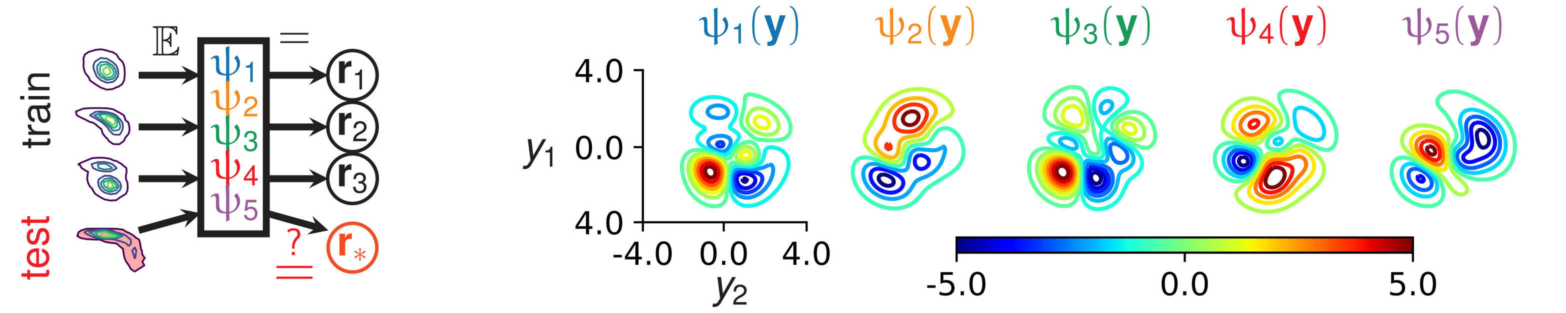

This suggests that the Bilinear network may be performing statistical computation.

- Does a DDC representation arise automatically in the Bilinear RNN?

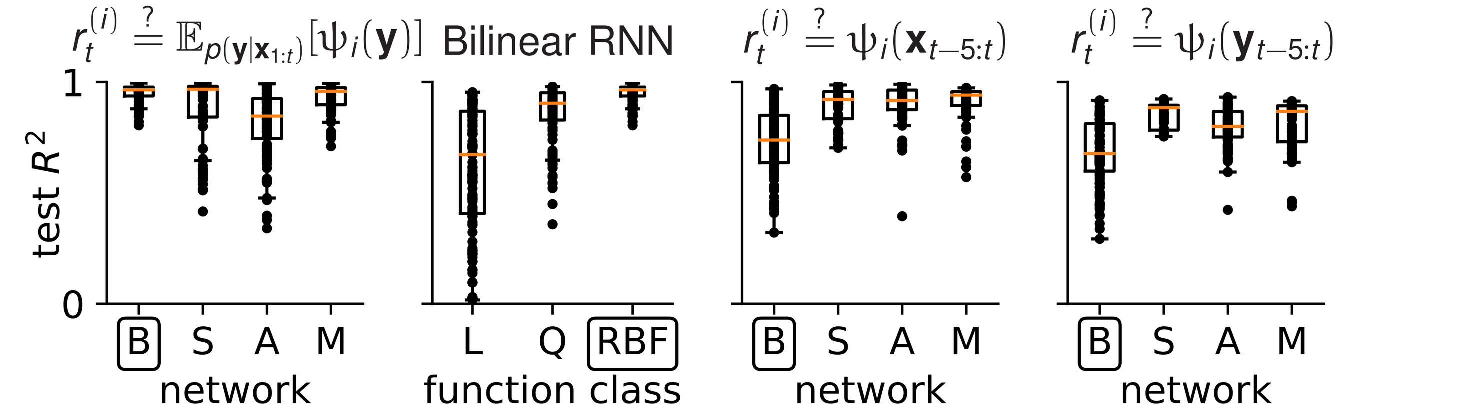
$$r_t^{(i)} \stackrel{?}{=} \mathbb{E}_{p(\mathbf{y}|\mathbf{x}_{1:t})}[\psi_i(\mathbf{y}_t)]$$

Use radial basis function (RBF) to approximate $\psi(\mathbf{y})$:

$$\psi(\mathbf{y}) := \sum_m \alpha_m k(\mathbf{y} - \mathbf{y}_m), \text{ where } k(\mathbf{y} - \mathbf{y}_m) = \exp\left(-\frac{\|\mathbf{y} - \mathbf{y}_m\|^2}{2\sigma^2}\right)$$




$\psi(\mathbf{y})$ found in the Bilinear network explains most of the variance in $\mathbf{r}$.
Using simpler $\psi(\mathbf{y})$ (linear or quadratic) or learning functions on data history yielded lower $R^2$.




## Conclusions

- The distributed distributional code (DDC) represents distributions by nonlinear “moments”.
- A neural network model with bilinear architecture consistent with this hypothesis, trained to perform a task requiring inference (but without explicit probabilistic supervision):
  - was more accurate than alternative networks
  - learnt consistent hidden activations that were nonlinear moments of the true posterior.
- Thus, the DDC offers a flexible, powerful and biologically plausible framework for **representation**, **computation** and **learning**
- Questions
  - How robust is this representation under noise?
  - How well can the network learn in an online fasion, back-propagating a short time window?
  - How to make the learning rule more biologically plausible?

[This work was supported by the Gatsby Charitable Foundation.]