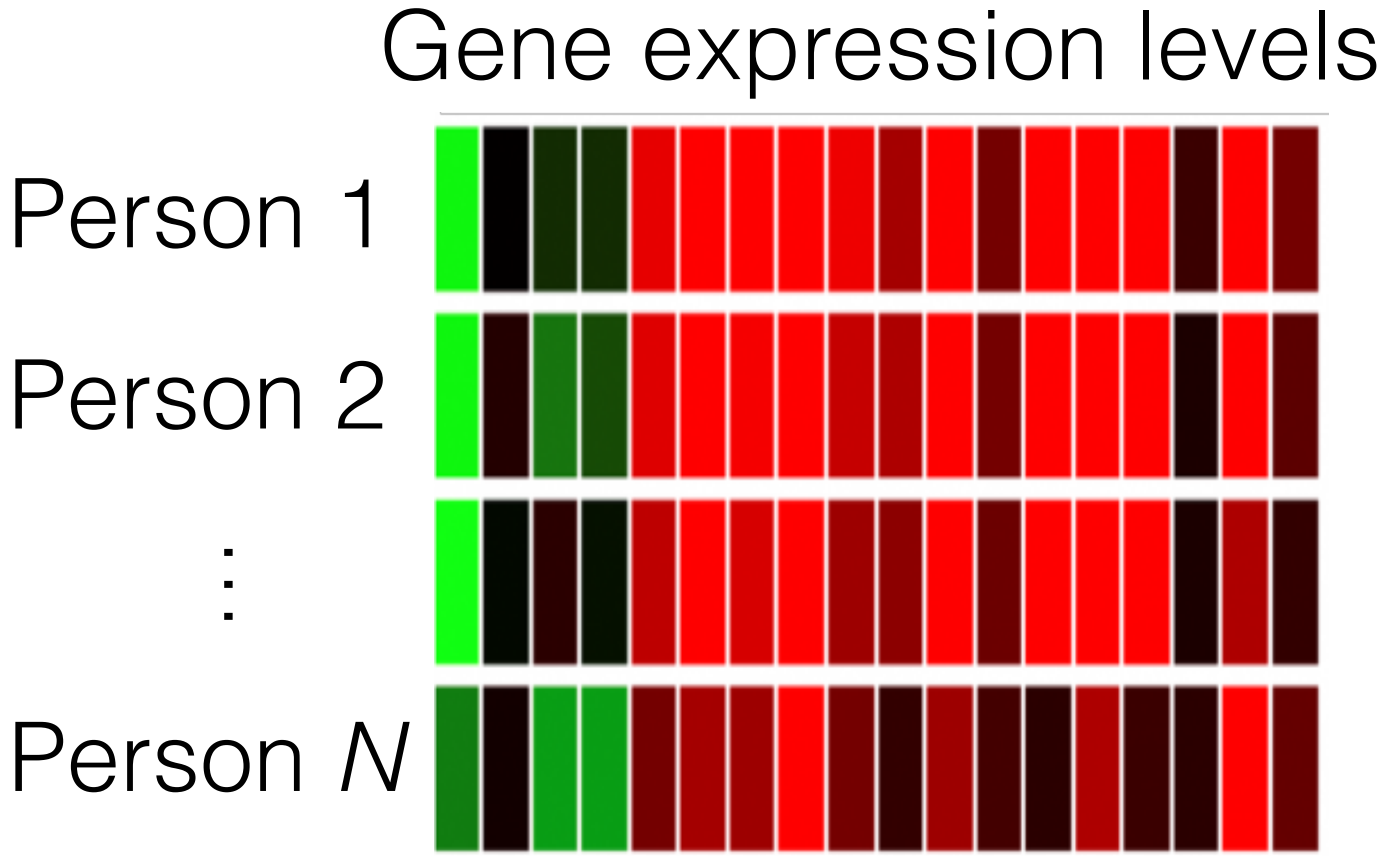
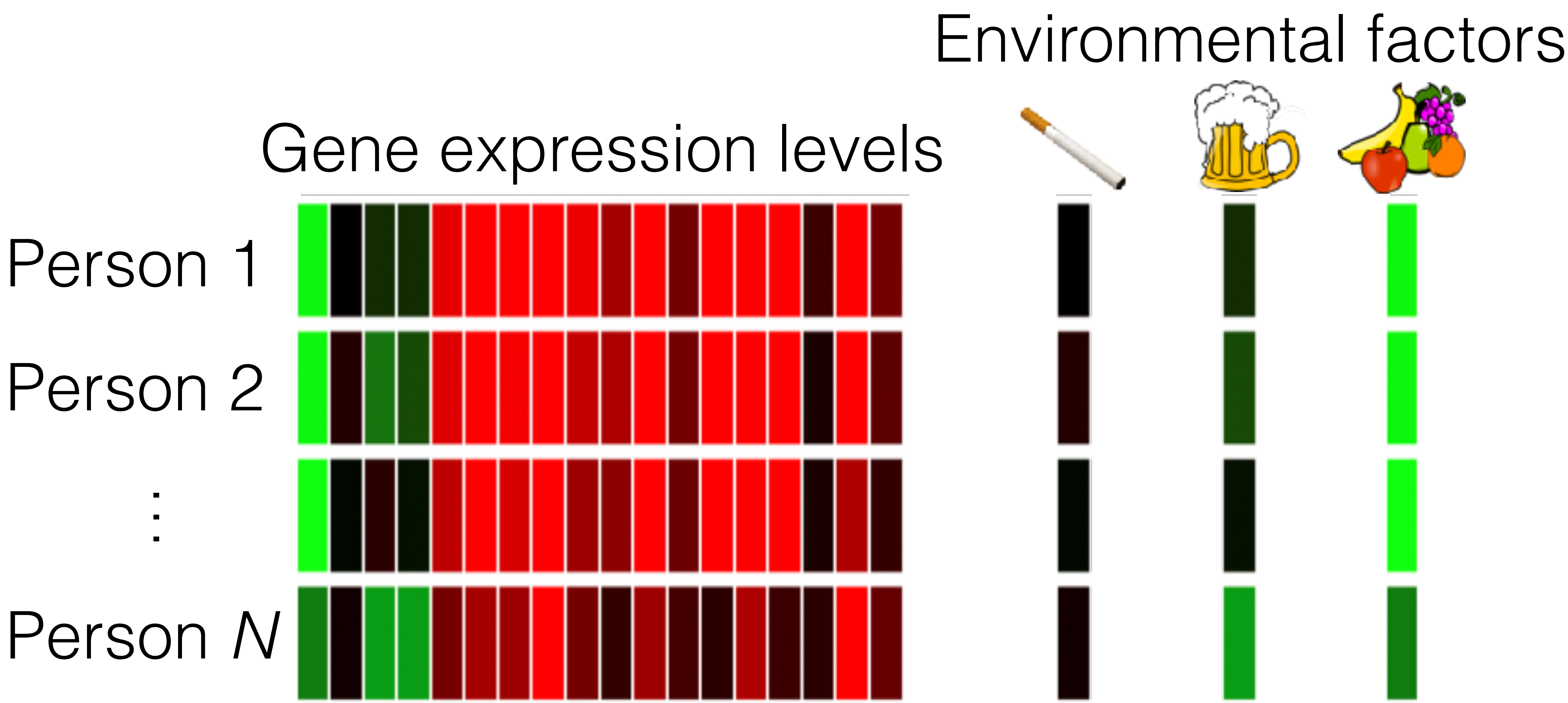
Fast Discovery of Pairwise Interactions in High Dimensions using Bayes

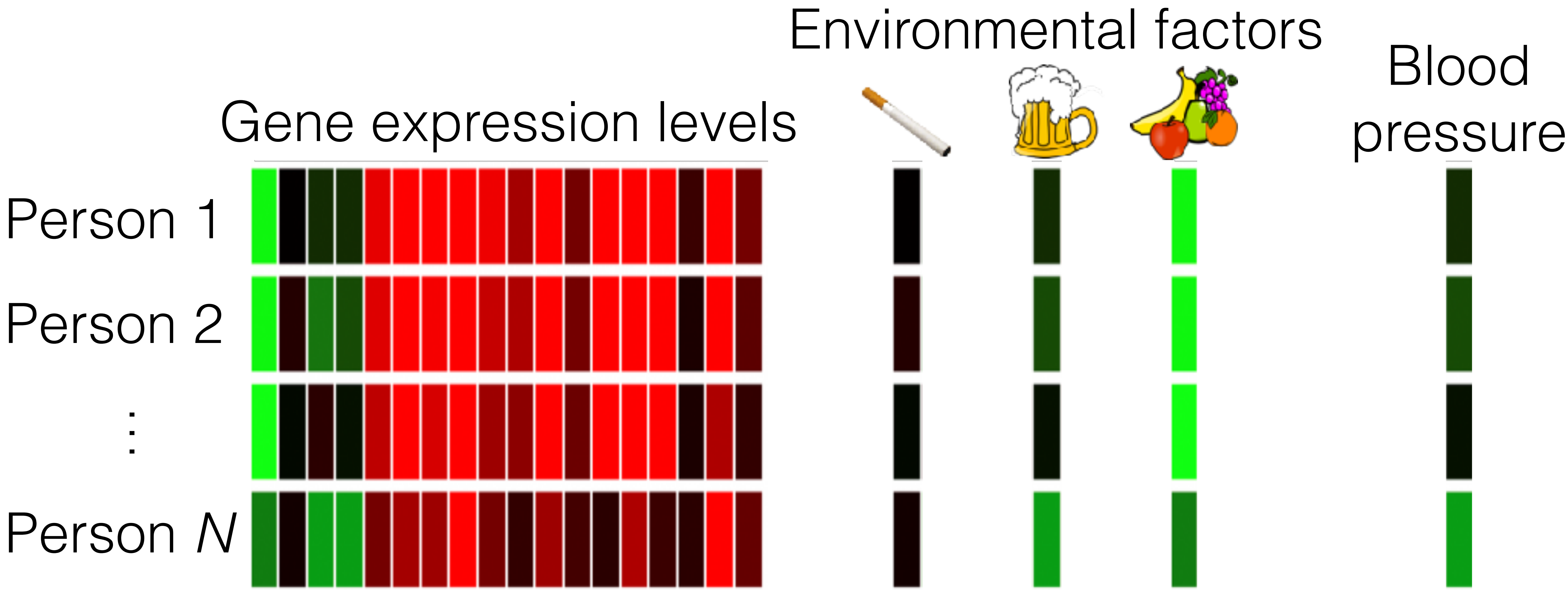
Tamara Broderick
Associate Professor
EECS, MIT

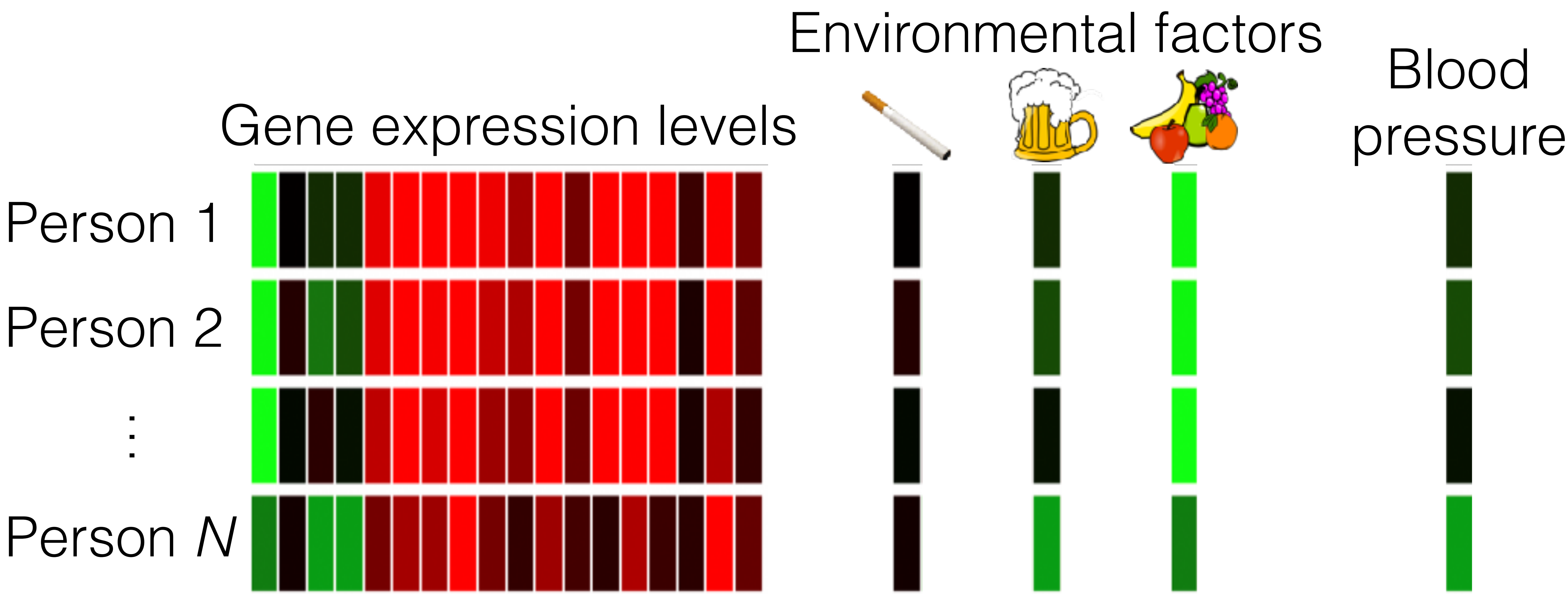
Raj Agrawal, Jonathan H. Huggins, Brian L. Trippe



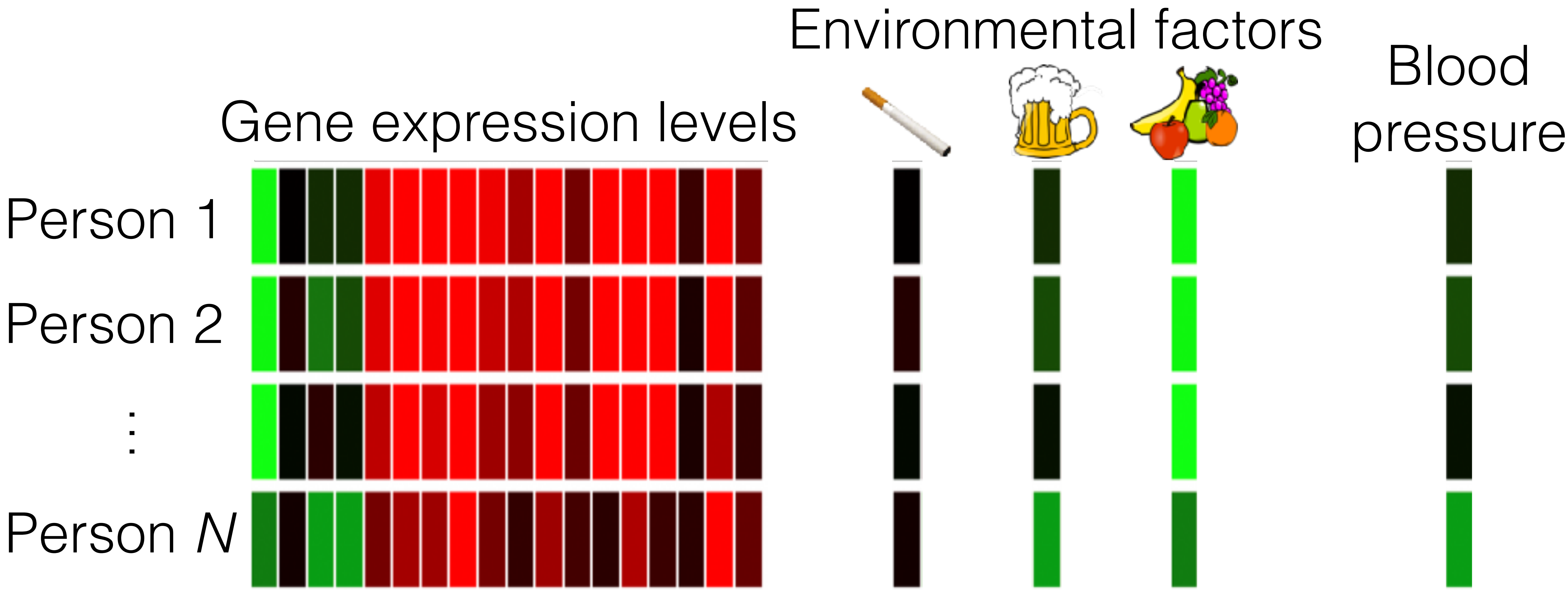

Gene expression levels
Person 1
Person 2
⋮
Person N


Environmental factors
Gene expression levels
Person 1
Person 2
⋮
Person N

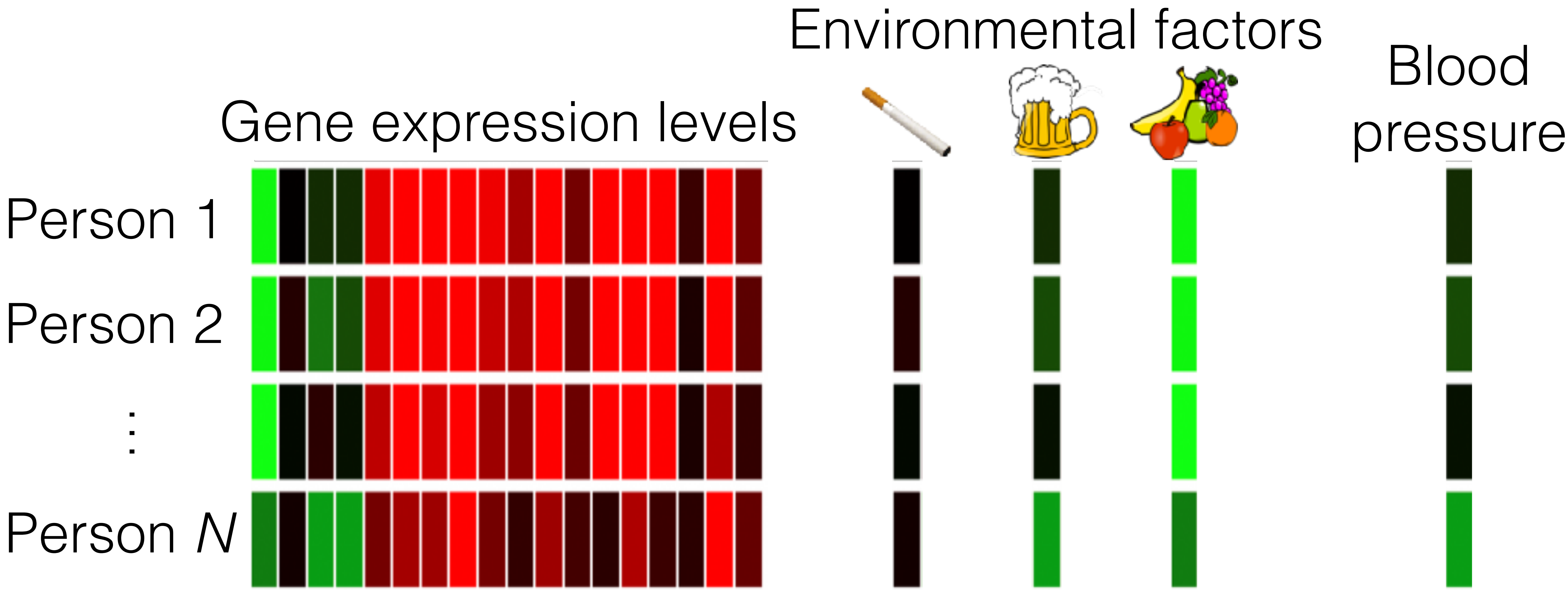

Environmental factors
Blood pressure
Gene expression levels
Person 1
Person 2
⋮
Person N



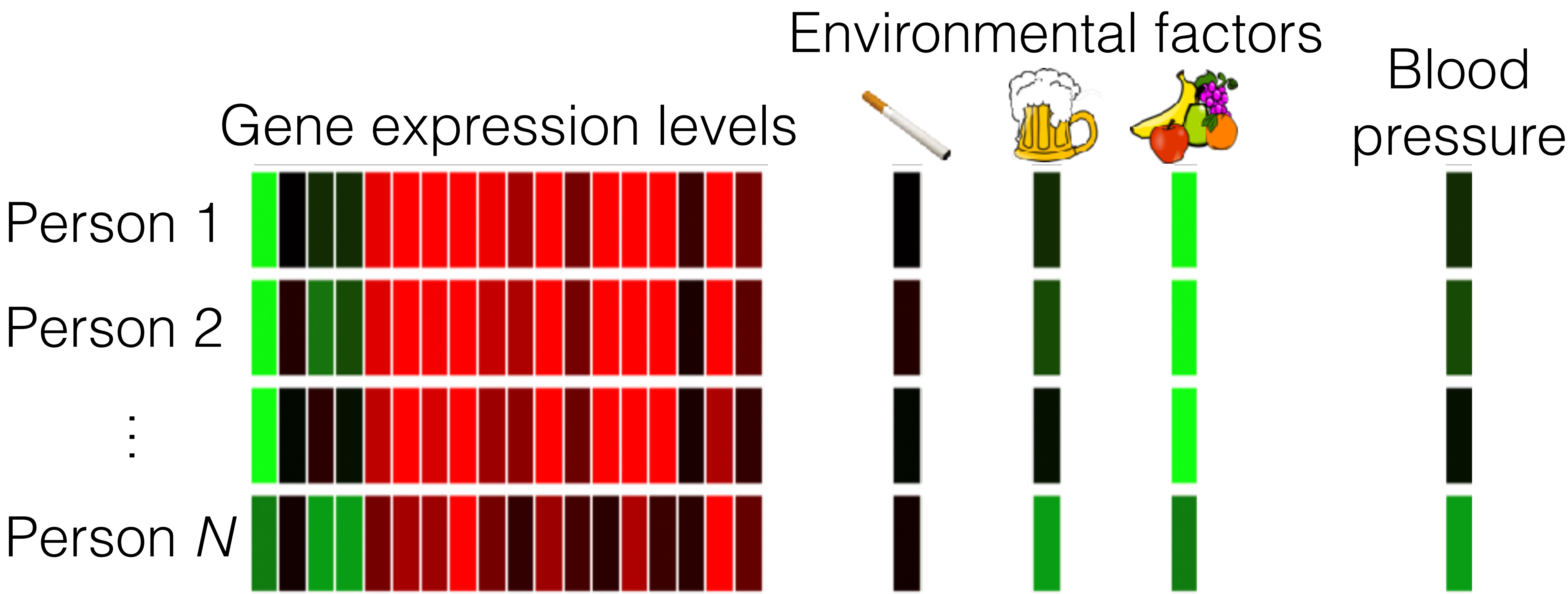
- Which genes/factors are associated with a health issue?



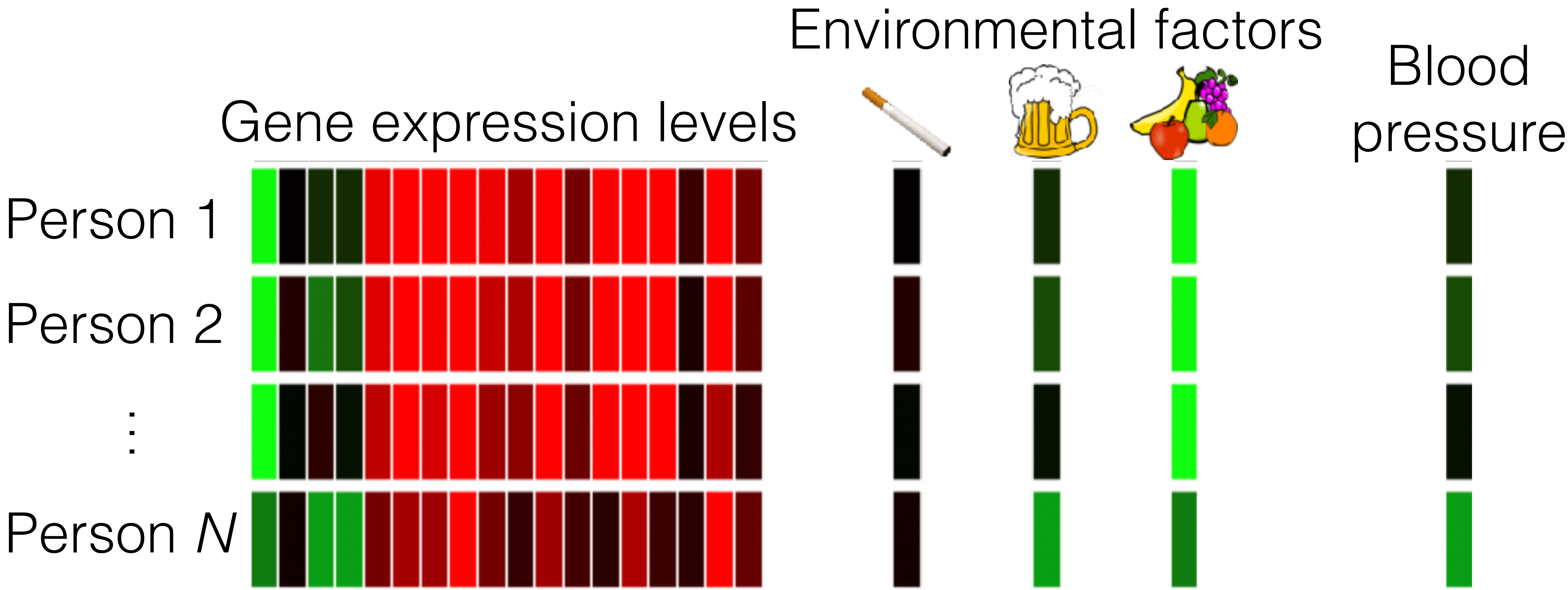
- Which genes/factors are associated with a health issue?
- Want small subset of p ($> N$) covariates



- Which genes/factors are associated with a health issue?
- Want small subset of p (> N) covariates (cf. LASSO)

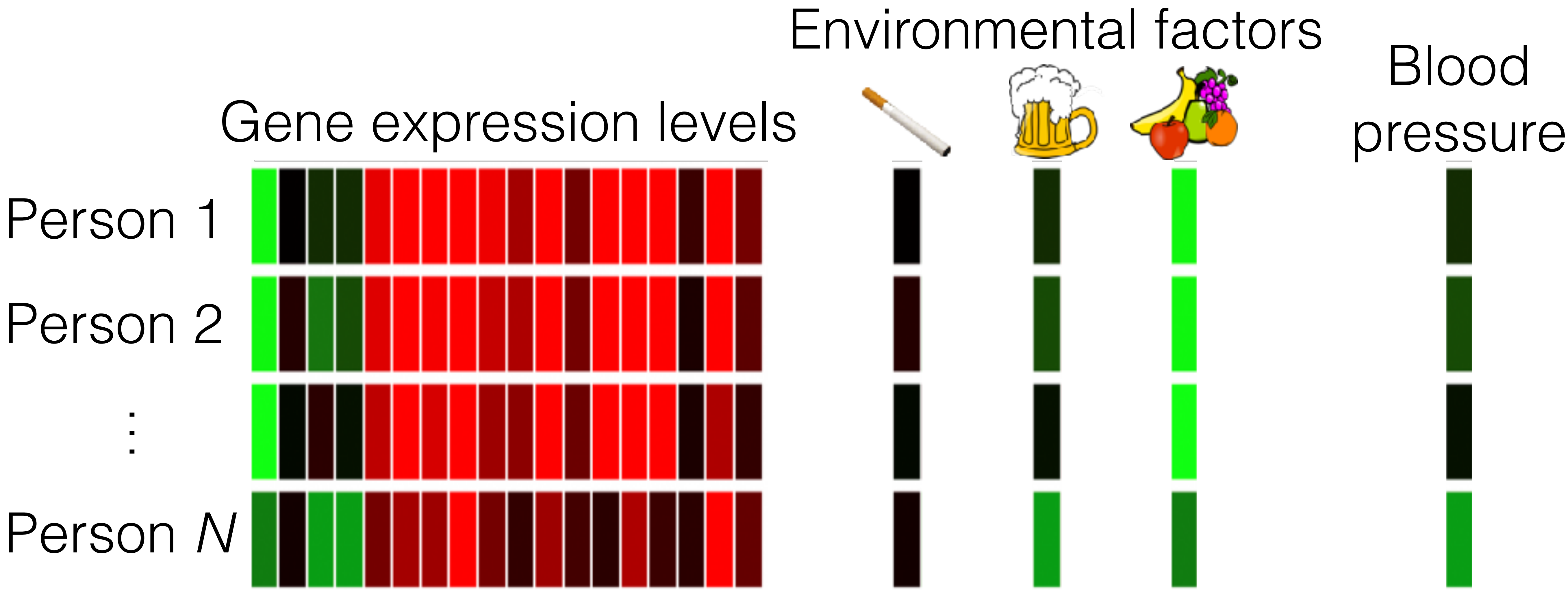


- Which genes/factors are associated with a health issue?
- Want small subset of p ($> N$) covariates (cf. LASSO)
- Additive model often not enough: need interactions



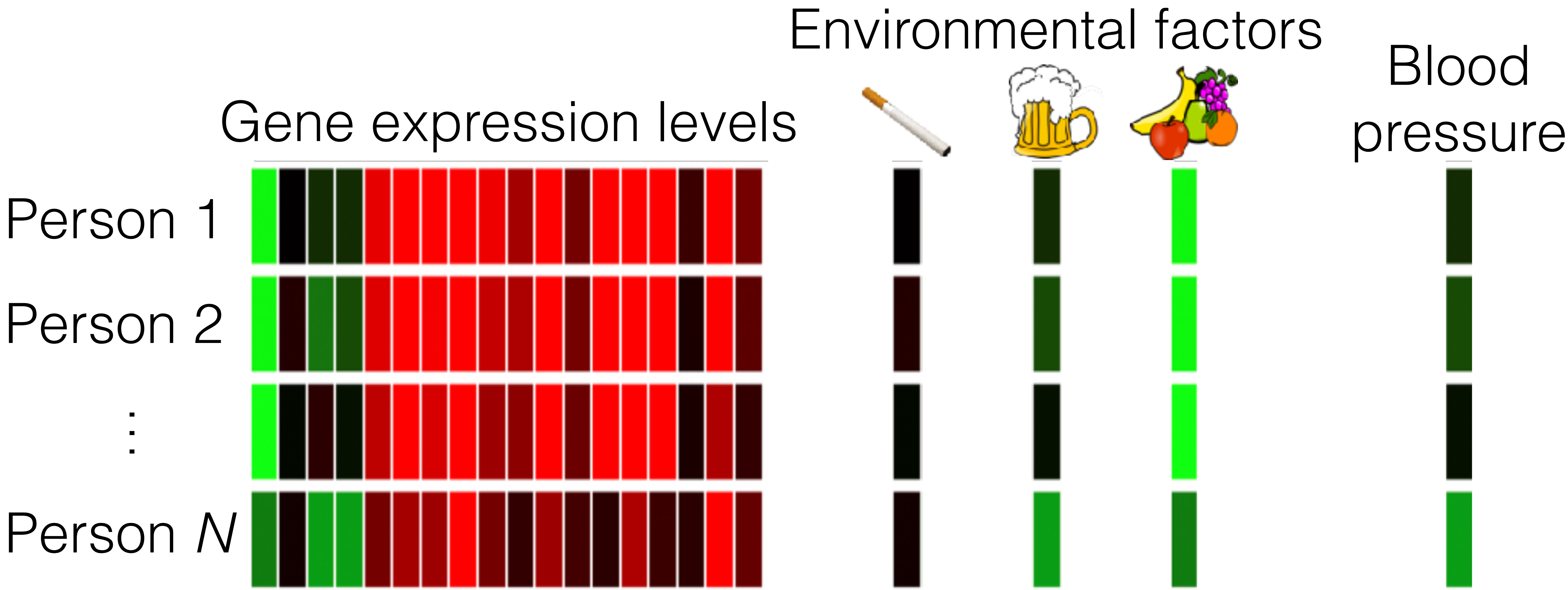
- Which genes/factors are associated with a health issue?
- Want small subset of p (> N) covariates (cf. LASSO)
- Additive model often not enough: need interactions (now p^2 dims!)

Pairwise interactions in high dimensions



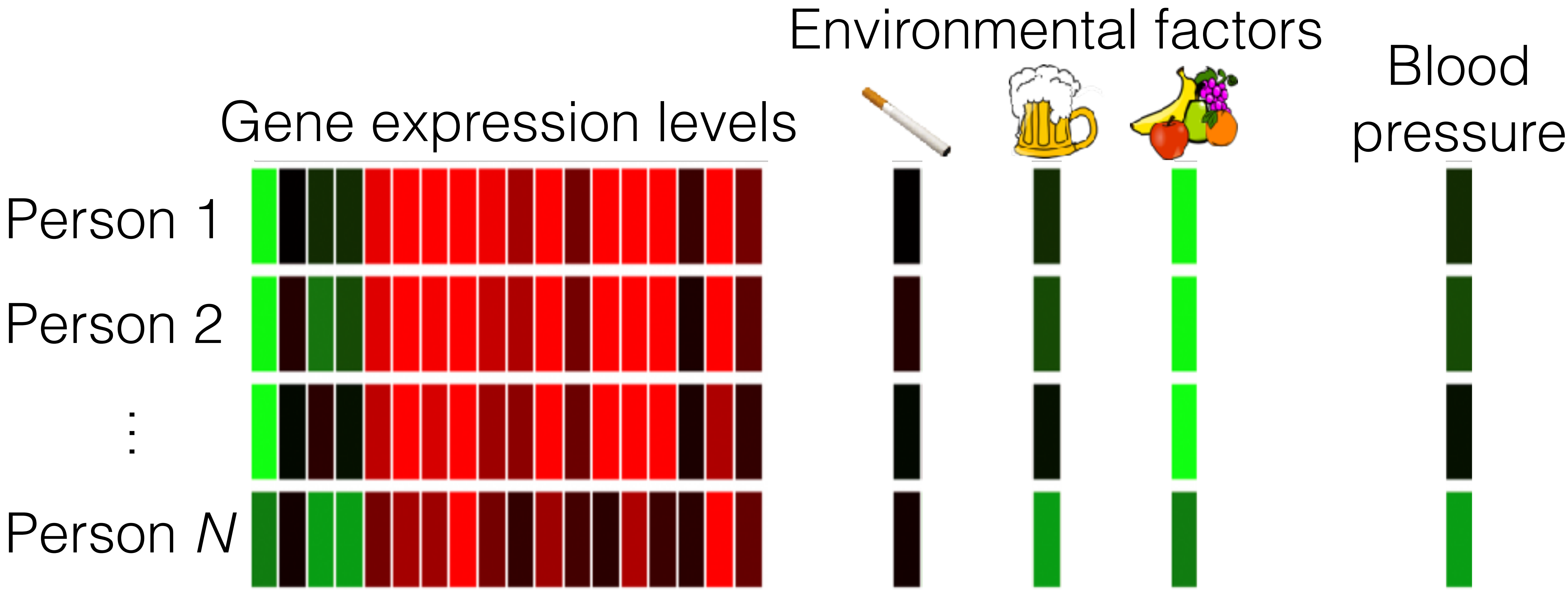
- Which genes/factors are associated with a health issue?
- Want small subset of p (> N) covariates (cf. LASSO)
- Additive model often not enough: need interactions (now p^2 dims!)

Pairwise interactions in high dimensions



- Which genes/factors are associated with a health issue?
- Want small subset of p (> N) covariates (cf. LASSO)
- Additive model often not enough: need interactions (now p^2 dims!)
- **We provide**: Fast, accurate (Bayes) method for interaction discovery

Pairwise interactions in high dimensions



- Which genes/factors are associated with a health issue?
- Want small subset of p ($> N$) covariates (cf. LASSO)
- Additive model often not enough: need interactions (now p^2 dims!)
- **We provide**: Fast, accurate (Bayes) method for interaction discovery
  - Better scaling in p & better accuracy than LASSO-based methods. Orders of magnitude faster than naive Bayesian inference

Roadmap

Roadmap

- Setup: Discovering main and interaction effects

Roadmap

- Setup: Discovering main and interaction effects
- Our method

Roadmap

- Setup: Discovering main and interaction effects
- Our method
  - A Bayesian generative model

Roadmap

- Setup: Discovering main and interaction effects
- Our method
  - A Bayesian generative model
  - Fast inference

Roadmap

- Setup: Discovering main and interaction effects
- Our method
  - A Bayesian generative model
  - Fast inference
  - Fast reporting of results

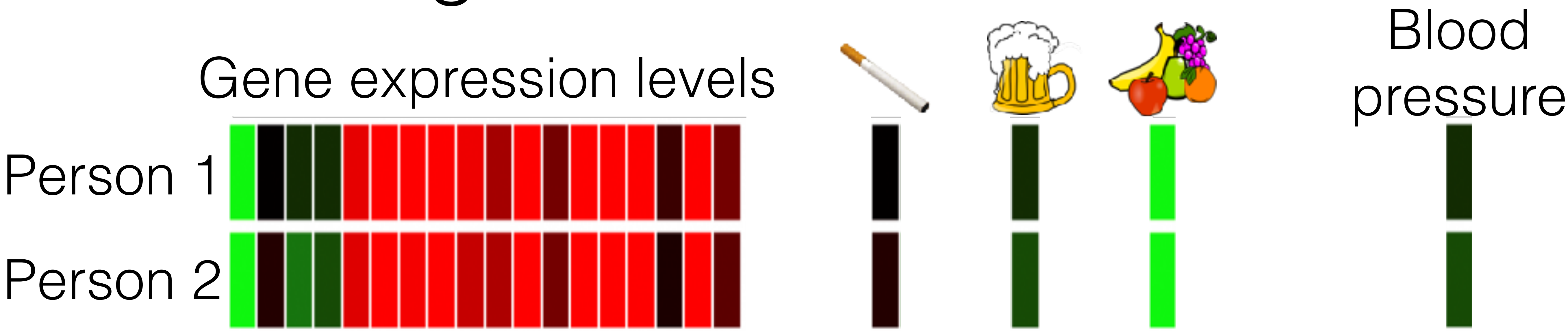
Roadmap

- Setup: Discovering main and interaction effects
- Our method
    - A Bayesian generative model
    - Fast inference
    - Fast reporting of results
- Experiments on simulated and real data

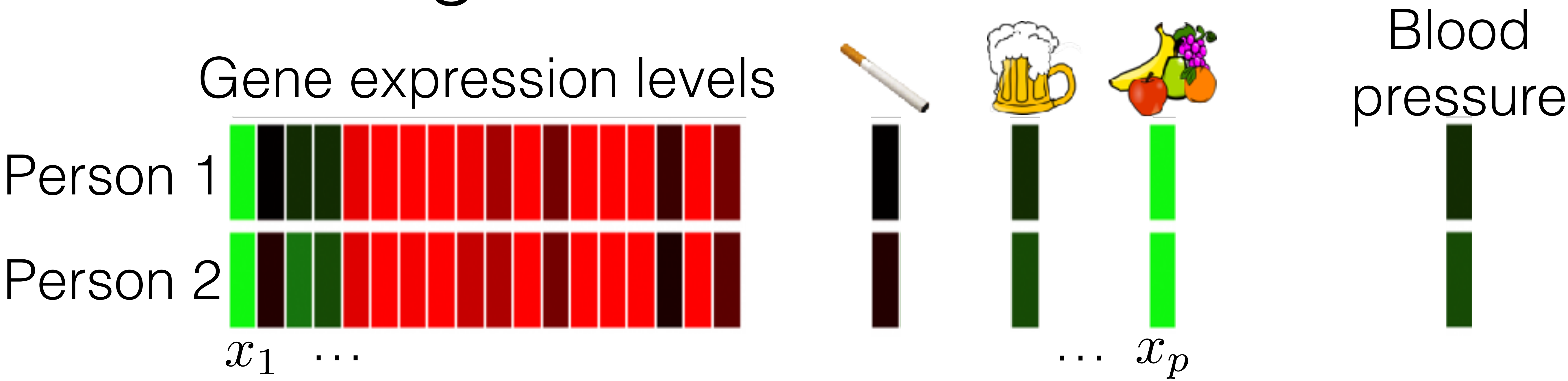
Roadmap

- Setup: Discovering main and interaction effects
- Our method
    - A Bayesian generative model
    - Fast inference
    - Fast reporting of results
- Experiments on simulated and real data

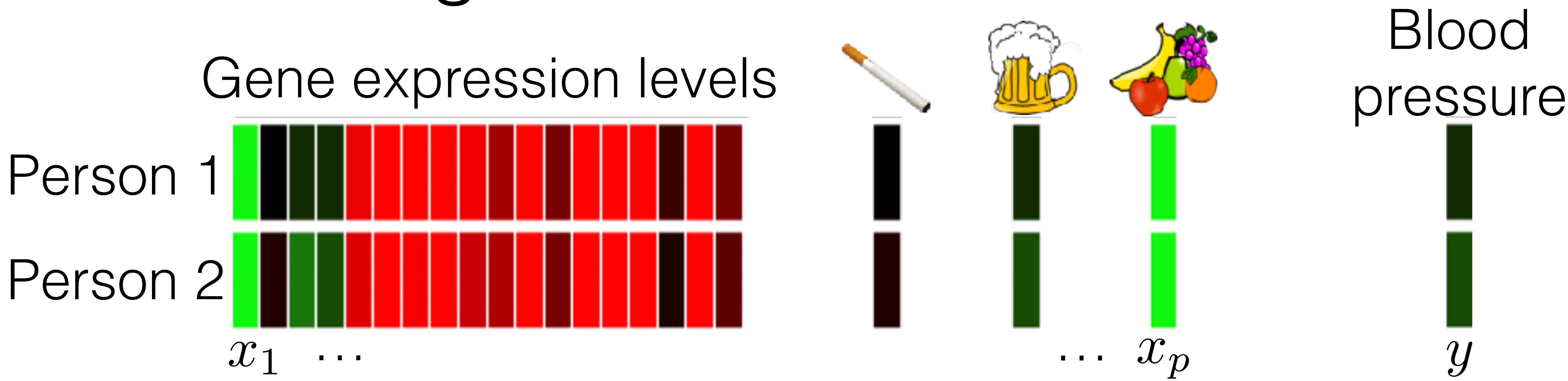
Discovering main and interaction effects



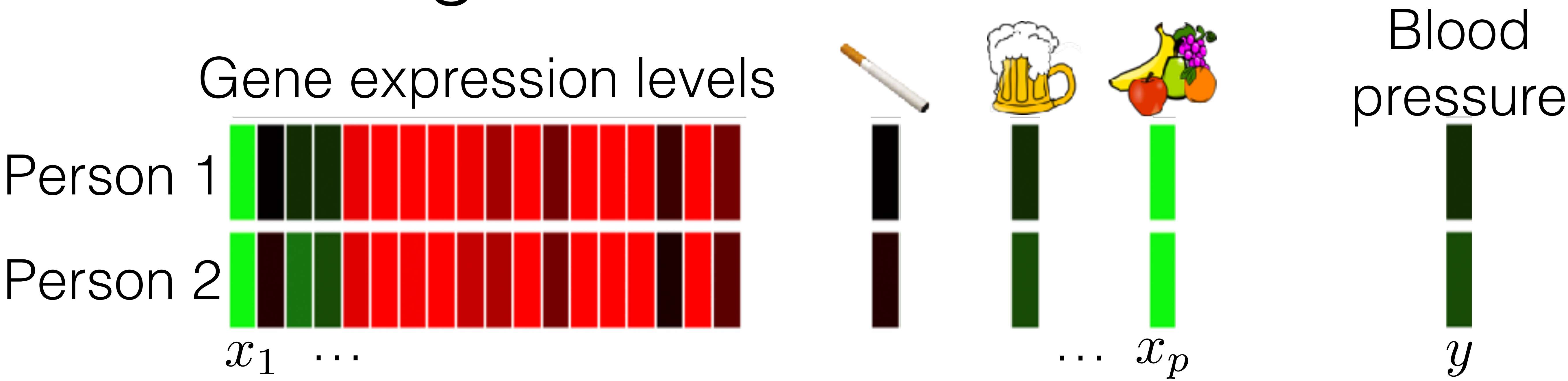
Discovering main and interaction effects



Discovering main and interaction effects

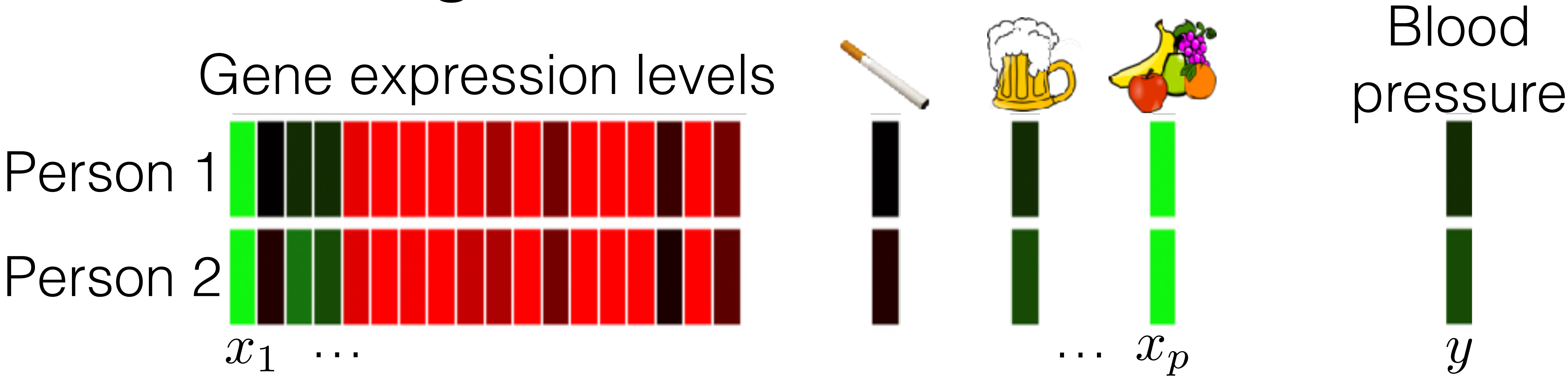


Discovering main and interaction effects



$$x^\top := [x_1, \ldots, x_p]$$

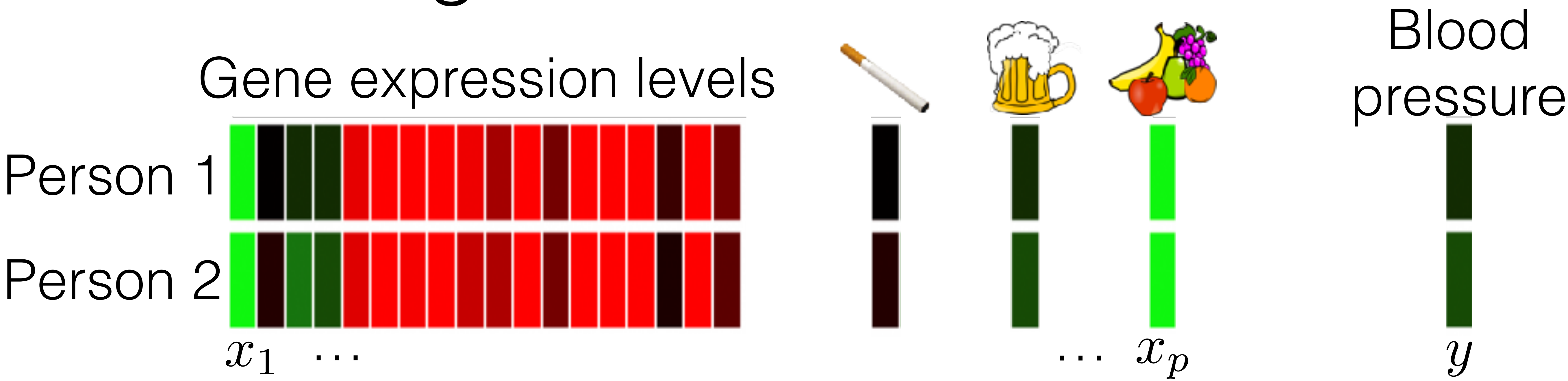
Discovering main and interaction effects



$$x^\top := [x_1, \ldots, x_p]$$

$$y^{(n)} = \theta^\top x^{(n)} + \epsilon^{(n)}, \quad \epsilon^{(n)} \overset{iid}{\sim} \mathcal{N}(0, \sigma^2)$$

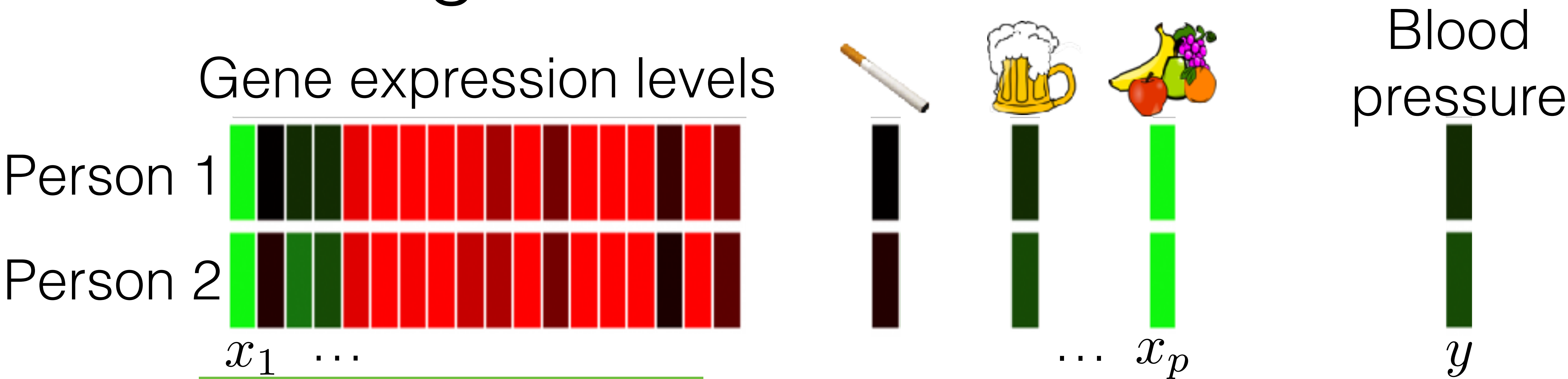
Discovering main and interaction effects



$$x^\top := [1, x_1, \ldots, x_p]$$

$$y^{(n)} = \theta^\top x^{(n)} + \epsilon^{(n)}, \quad \epsilon^{(n)} \overset{iid}{\sim} \mathcal{N}(0, \sigma^2)$$

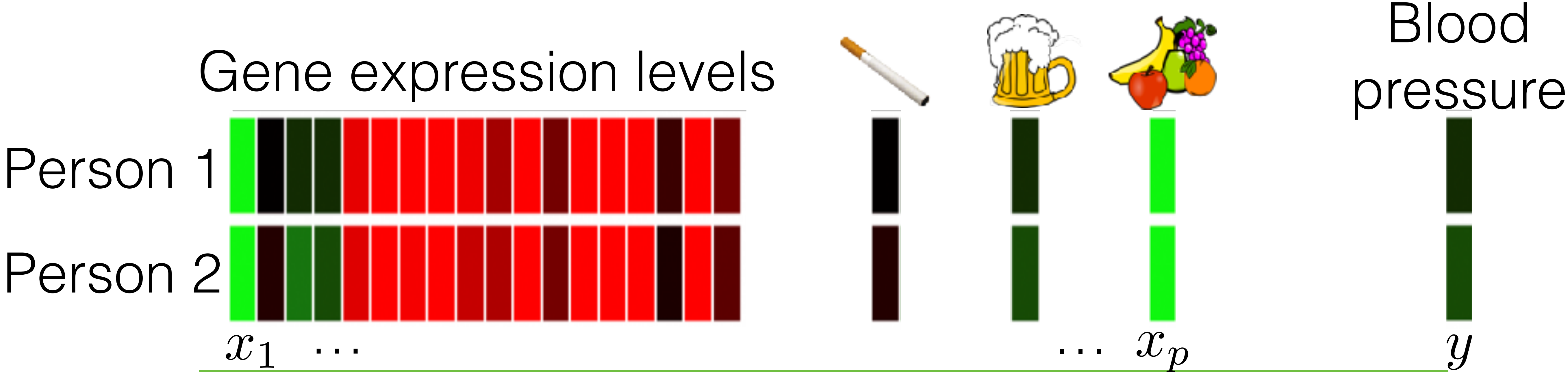
Discovering main and interaction effects



$$x^\top := [1, x_1, \ldots, x_p]$$

$$y^{(n)} = \theta^\top x^{(n)} + \epsilon^{(n)}, \quad \epsilon^{(n)} \overset{iid}{\sim} \mathcal{N}(0, \sigma^2)$$

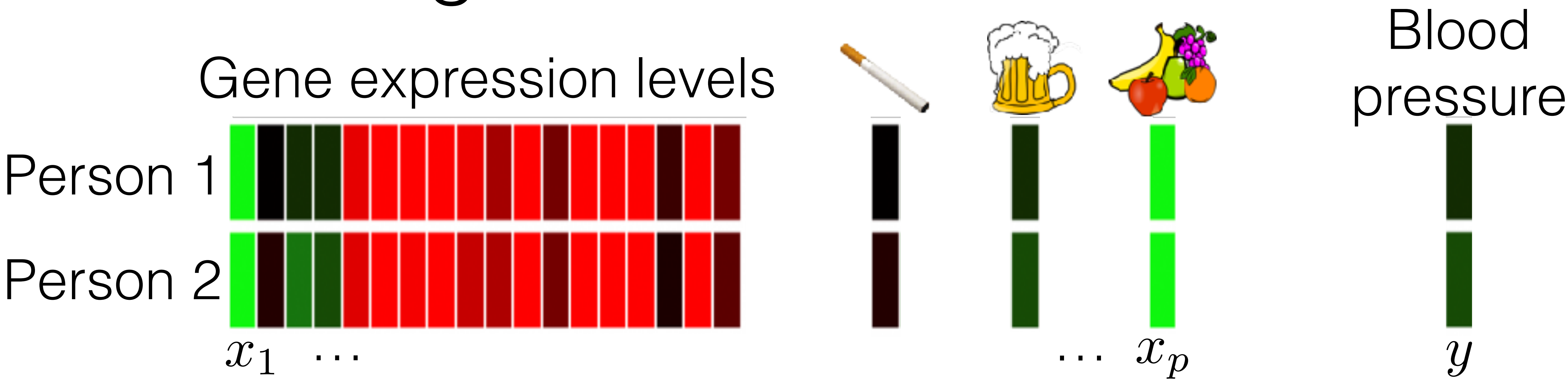
Discovering main and interaction effects



$$\Phi_2^\top(x) := [1, x_1, \ldots, x_p, x_1 x_2, \ldots, x_{p-1} x_p, x_1^2, \ldots, x_p^2]$$

$$y^{(n)} = \theta^\top x^{(n)} + \epsilon^{(n)}, \quad \epsilon^{(n)} \overset{iid}{\sim} \mathcal{N}(0, \sigma^2)$$

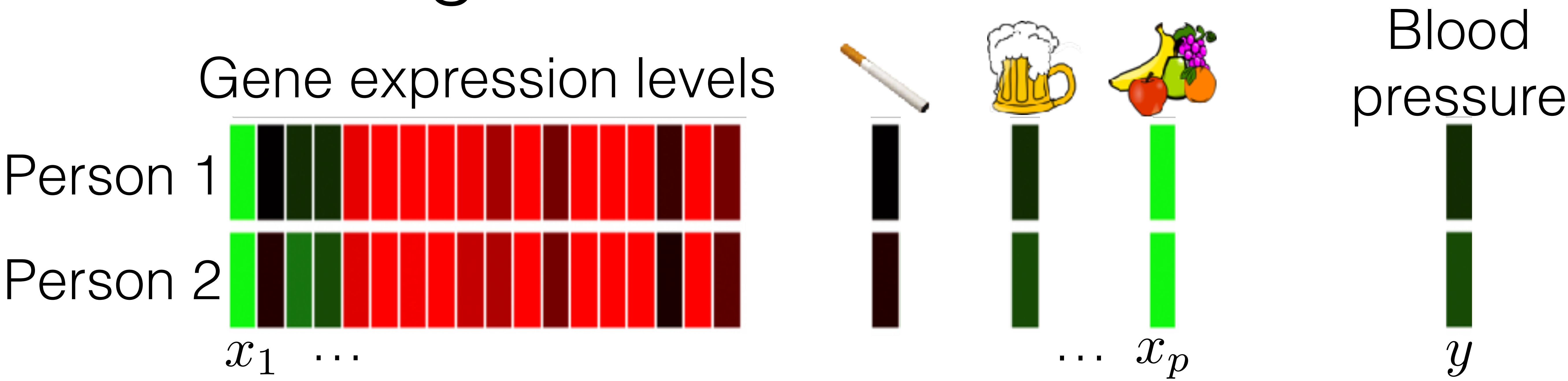
Discovering main and interaction effects



$$\Phi_2^\top(x) := [1, x_1, \ldots, x_p, x_1 x_2, \ldots, x_{p-1} x_p, x_1^2, \ldots, x_p^2]$$

$$y^{(n)} = \theta^\top x^{(n)} + \epsilon^{(n)}, \quad \epsilon^{(n)} \overset{iid}{\sim} \mathcal{N}(0, \sigma^2)$$

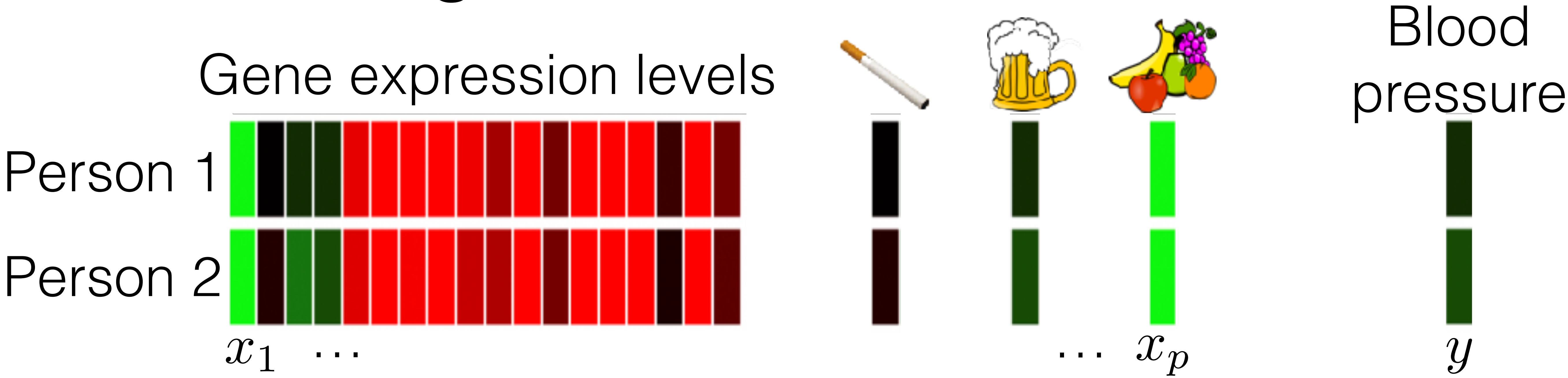
Discovering main and interaction effects



$$\Phi_2^\top(x) := [1, x_1, \ldots, x_p, x_1 x_2, \ldots, x_{p-1} x_p, x_1^2, \ldots, x_p^2]$$

$$y^{(n)} = \theta^\top \Phi_2(x^{(n)}) + \epsilon^{(n)}, \quad \epsilon^{(n)} \overset{iid}{\sim} \mathcal{N}(0, \sigma^2)$$

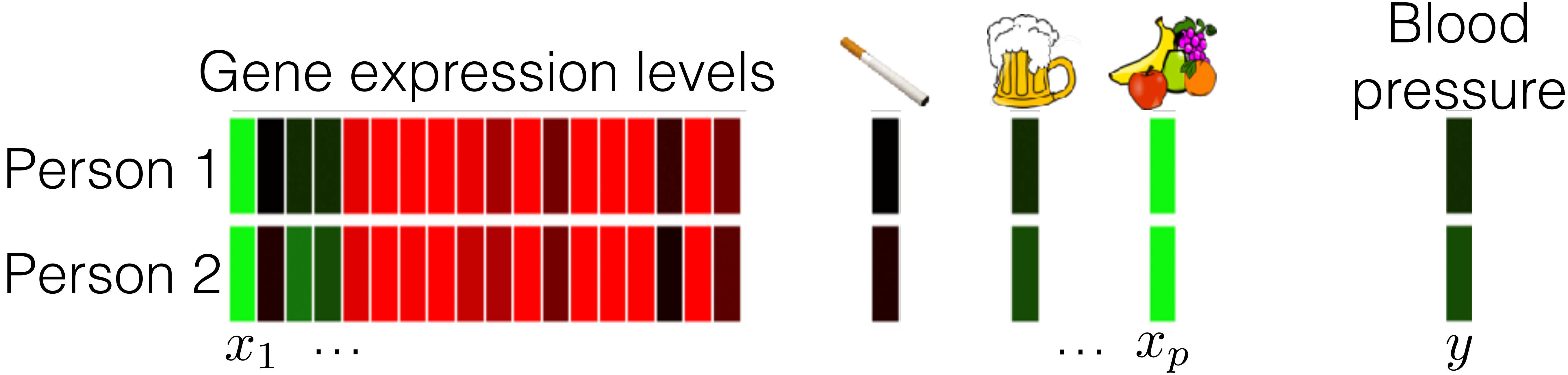
Discovering main and interaction effects



$$\Phi_2^\top(x) := [1, x_1, \ldots, x_p, x_1 x_2, \ldots, x_{p-1} x_p, x_1^2, \ldots, x_p^2]$$

$$y^{(n)} = \theta^\top \Phi_2(x^{(n)}) + \epsilon^{(n)}, \quad \epsilon^{(n)} \overset{iid}{\sim} \mathcal{N}(0, \sigma^2)$$

Discovering main and interaction effects

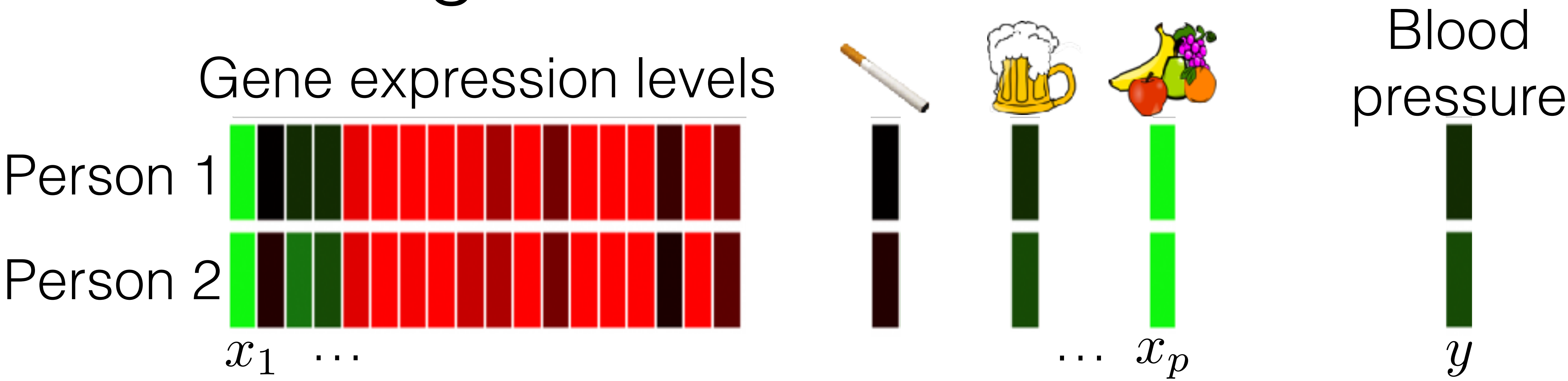


$$\Phi_2^\top(x) := [1, x_1, \ldots, x_p, x_1 x_2, \ldots, x_{p-1} x_p, x_1^2, \ldots, x_p^2]$$

$$y^{(n)} = \theta^\top \Phi_2(x^{(n)}) + \epsilon^{(n)}, \quad \epsilon^{(n)} \overset{iid}{\sim} \mathcal{N}(0, \sigma^2)$$

- **Goal**: Parameter selection/estimation

Discovering main and interaction effects

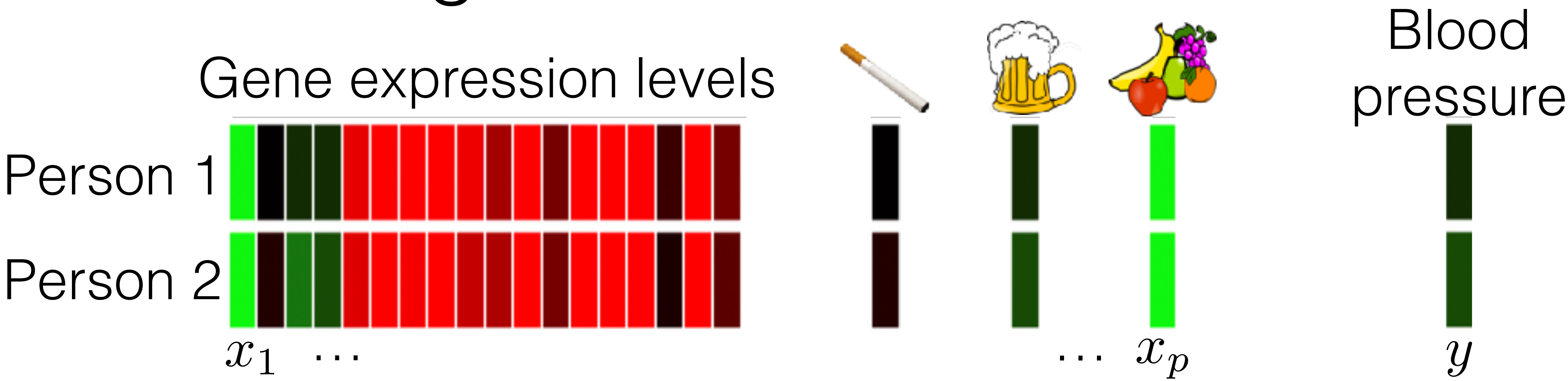


$$\Phi_2^\top(x) := [1, x_1, \ldots, x_p, x_1 x_2, \ldots, x_{p-1} x_p, x_1^2, \ldots, x_p^2]$$

$$y^{(n)} = \theta^\top \Phi_2(x^{(n)}) + \epsilon^{(n)}, \quad \epsilon^{(n)} \overset{iid}{\sim} \mathcal{N}(0, \sigma^2)$$

- **Goal**: Parameter selection/estimation under assumptions:

Discovering main and interaction effects

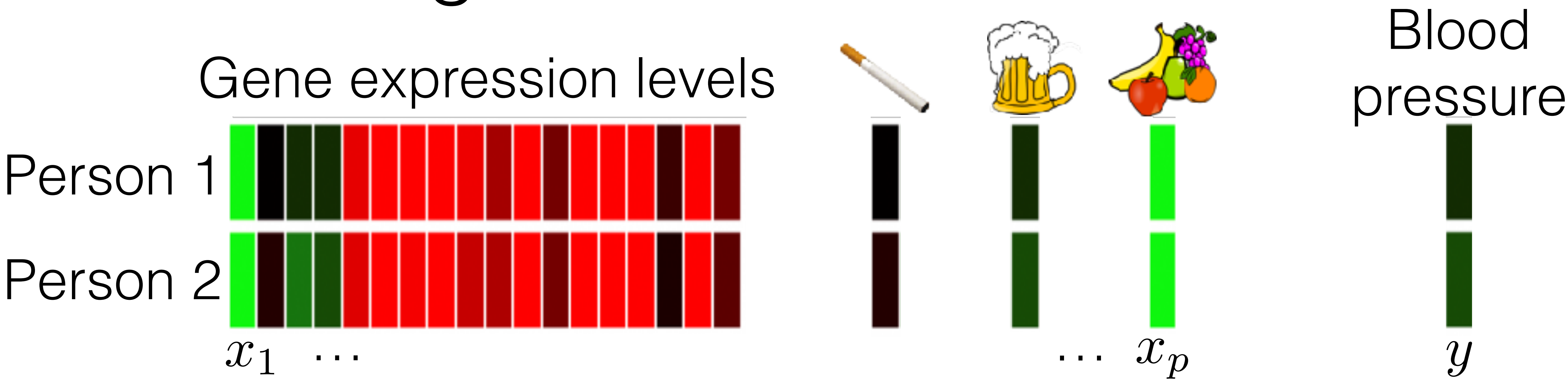


$$\Phi_2^\top(x) := [1, x_1, \ldots, x_p, x_1 x_2, \ldots, x_{p-1} x_p, x_1^2, \ldots, x_p^2]$$

$$y^{(n)} = \theta^\top \Phi_2(x^{(n)}) + \epsilon^{(n)}, \quad \epsilon^{(n)} \overset{iid}{\sim} \mathcal{N}(0, \sigma^2)$$

- **Goal**: Parameter selection/estimation under assumptions:
  - *Sparsity*: most main effects are negligible (interpretable)

Discovering main and interaction effects



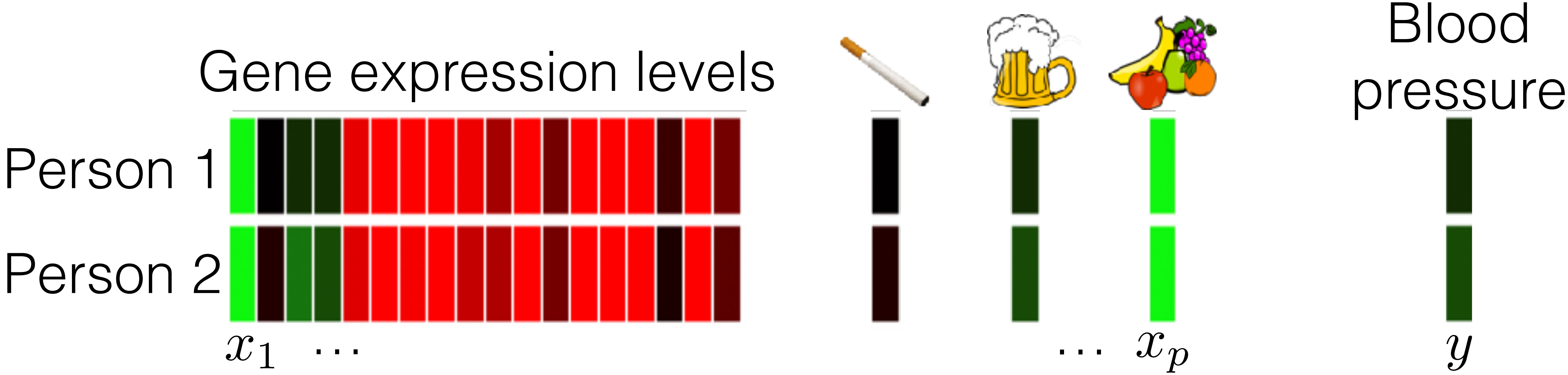
$$\Phi_2^\top(x) := [1, x_1, \ldots, x_p, x_1 x_2, \ldots, x_{p-1} x_p, x_1^2, \ldots, x_p^2]$$

$$y^{(n)} = \theta^\top \Phi_2(x^{(n)}) + \epsilon^{(n)}, \quad \epsilon^{(n)} \overset{iid}{\sim} \mathcal{N}(0, \sigma^2)$$

- **Goal**: Parameter selection/estimation under assumptions:
  - *Sparsity*: most main effects are negligible (interpretable)
  - *Strong hierarchy*: Interaction only if main effects are present

[Chipman 1996; Wu et al 2009; Bien et al 2013; Lim, Hastie 2015; Nakagawa et al 2016; Griffin, Brown 2017]

Discovering main and interaction effects



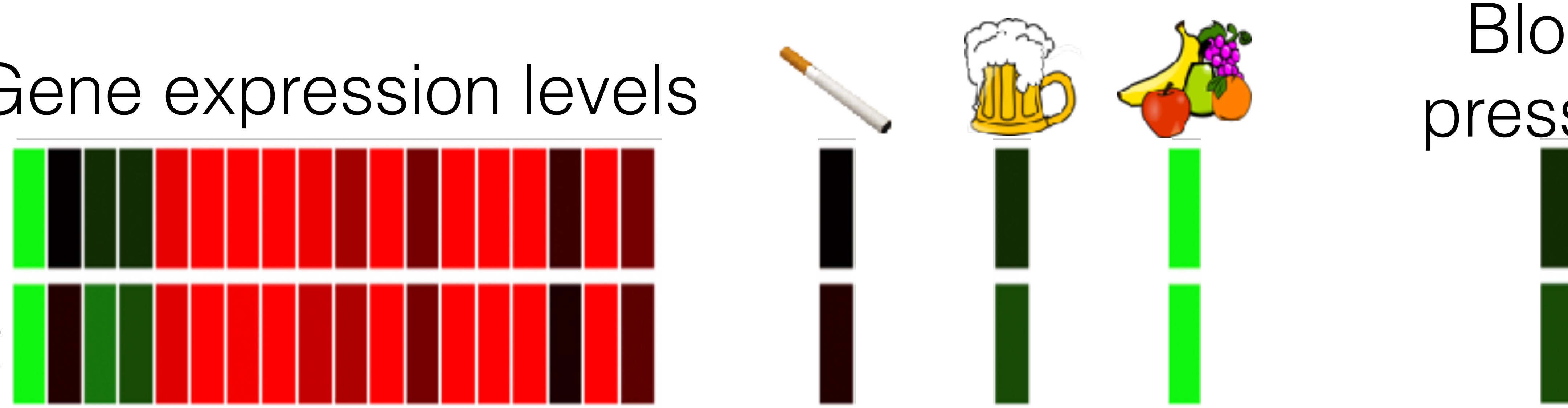
$$\Phi_2^\top(x) := [1, x_1, \ldots, x_p, x_1 x_2, \ldots, x_{p-1} x_p, x_1^2, \ldots, x_p^2]$$

$$y^{(n)} = \theta^\top \Phi_2(x^{(n)}) + \epsilon^{(n)}, \quad \epsilon^{(n)} \overset{iid}{\sim} \mathcal{N}(0, \sigma^2)$$

- **Goal**: Parameter selection/estimation under assumptions:
  - *Sparsity*: most main effects are negligible (interpretable)
  - *Strong hierarchy*: Interaction only if main effects are present
- p^2 covariates: large p ➔ statistical & computational challenge

[Chipman 1996; Wu et al 2009; Bien et al 2013; Lim, Hastie 2015; Nakagawa et al 2016; Griffin, Brown 2017]

Discovering main and interaction effects



$$\Phi_2^\top(x) := [1, x_1, \ldots, x_p, x_1 x_2, \ldots, x_{p-1} x_p, x_1^2, \ldots, x_p^2]$$

$$y^{(n)} = \theta^\top \Phi_2(x^{(n)}) + \epsilon^{(n)}, \quad \epsilon^{(n)} \overset{iid}{\sim} \mathcal{N}(0, \sigma^2)$$

- **Goal**: Parameter selection/estimation under assumptions:
  - *Sparsity*: most main effects are negligible (interpretable)
  - *Strong hierarchy*: Interaction only if main effects are present
- p^2 covariates: large p ➔ statistical & computational challenge
- **Our solution**: using structure in covariates + sparsity assumptions to reduce to a problem *linear* in p

 [Chipman 1996; Wu et al 2009; Bien et al 2013; Lim, Hastie 2015; Nakagawa et al 2016; Griffin, Brown 2017]

Roadmap

- Setup: Discovering main and interaction effects
- Our method
  - A Bayesian generative model
  - Fast inference
  - Fast reporting of results
- Experiments on simulated and real data

Roadmap

- Setup: Discovering main and interaction effects
- Our method
  - A Bayesian generative model
  - Fast inference
  - Fast reporting of results
- Experiments on simulated and real data

Our approach

A Bayesian method: expert information, uncertainty quantification, regularization

Our approach

A Bayesian method: expert information, uncertainty quantification, regularization

1. Choose generative model

Our approach

A Bayesian method: expert information, uncertainty quantification, regularization

1. Choose generative model

2. Compute posterior

Our approach

A Bayesian method: expert information, uncertainty quantification, regularization

1. Choose generative model

2. Compute posterior

3. Report relevant summaries of the posterior

Our approach

A Bayesian method: expert information, uncertainty quantification, regularization

1. Choose generative model

2. Compute posterior

3. Report relevant summaries of the posterior

Our approach

A Bayesian method: expert information, uncertainty quantification, regularization

1. New Bayesian generative model: **Sparse Kernel Interaction Model (SKIM)** to encode sparsity and strong hierarchy
2. Compute posterior
3. Report relevant summaries of the posterior

Our approach

A Bayesian method: expert information, uncertainty quantification, regularization

1. New Bayesian generative model: **Sparse Kernel Interaction Model (SKIM)** to encode sparsity and strong hierarchy [Carvalho et al 2009; Piironen, Vehtari 2017; Chipman 1996, Griffin & Brown 2017]
2. Compute posterior
3. Report relevant summaries of the posterior

Our approach

A Bayesian method: expert information, uncertainty quantification, regularization

1. New Bayesian generative model: **Sparse Kernel Interaction Model (SKIM)** to encode sparsity and strong hierarchy [Carvalho et al 2009; Piironen, Vehtari 2017; Chipman 1996, Griffin & Brown 2017]
2. Compute posterior
3. Report relevant summaries of the posterior

Our approach

A Bayesian method: expert information, uncertainty quantification, regularization

1. New Bayesian generative model: **<u>S</u>parse <u>K</u>ernel <u>I</u>nteraction <u>M</u>odel (SKIM)** to encode sparsity and strong hierarchy [Carvalho et al 2009; Piironen, Vehtari 2017; Chipman 1996, Griffin & Brown 2017]
2. **<u>K</u>ernel <u>I</u>nteraction <u>S</u>ampler (KIS)**: Use kernel trick to run MCMC in $O(p)$ time per iteration
3. Report relevant summaries of the posterior

Our approach

A Bayesian method: expert information, uncertainty quantification, regularization

1. New Bayesian generative model: **Sparse Kernel Interaction Model (SKIM)** to encode sparsity and strong hierarchy [Carvalho et al 2009; Piironen, Vehtari 2017; Chipman 1996, Griffin & Brown 2017]
2. **Kernel Interaction Sampler (KIS)**: Use kernel trick to run MCMC in $O(p)$ time per iteration
3. Report relevant summaries of the posterior

Our approach

A Bayesian method: expert information, uncertainty quantification, regularization

1. New Bayesian generative model: **Sparse Kernel Interaction Model (SKIM)** to encode sparsity and strong hierarchy [Carvalho et al 2009; Piironen, Vehtari 2017; Chipman 1996, Griffin & Brown 2017]
2. **Kernel Interaction Sampler (KIS)**: Use kernel trick to run MCMC in $O(p)$ time per iteration
3. **Kernel Interaction Trick (KIT)**: Use kernel trick to report *all* non-negligible main and interaction effects in $O(p)$ time

Our approach

A Bayesian method: expert information, uncertainty quantification, regularization

1. New Bayesian generative model: **<u>S</u>parse <u>K</u>ernel <u>I</u>nteraction <u>M</u>odel (SKIM)** to encode sparsity and strong hierarchy [Carvalho et al 2009; Piironen, Vehtari 2017; Chipman 1996, Griffin & Brown 2017]
2. **<u>K</u>ernel <u>I</u>nteraction <u>S</u>ampler (KIS)**: Use kernel trick to run MCMC in $O(p)$ time per iteration
3. **<u>K</u>ernel <u>I</u>nteraction <u>T</u>rick (KIT)**: Use kernel trick to report *all* non-negligible main and interaction effects in $O(p)$ time

Our approach

A Bayesian method: expert information, uncertainty quantification, regularization

1. New Bayesian generative model: **<u>S</u>parse <u>K</u>ernel <u>I</u>nteraction <u>M</u>odel (SKIM)** to encode sparsity and strong hierarchy [Carvalho et al 2009; Piironen, Vehtari 2017; Chipman 1996, Griffin & Brown 2017]
2. **<u>K</u>ernel <u>I</u>nteraction <u>S</u>ampler (KIS)**: Use kernel trick to run MCMC in $O(p)$ time per iteration
3. **<u>K</u>ernel <u>I</u>nteraction <u>T</u>rick (KIT)**: Use kernel trick to report *all* non-negligible main and interaction effects in $O(p)$ time

Not just for SKIM

Kernel Interaction Sampler vs. Naive MCMC

- MCMC option 1: sample θ

Kernel Interaction Sampler vs. Naive MCMC

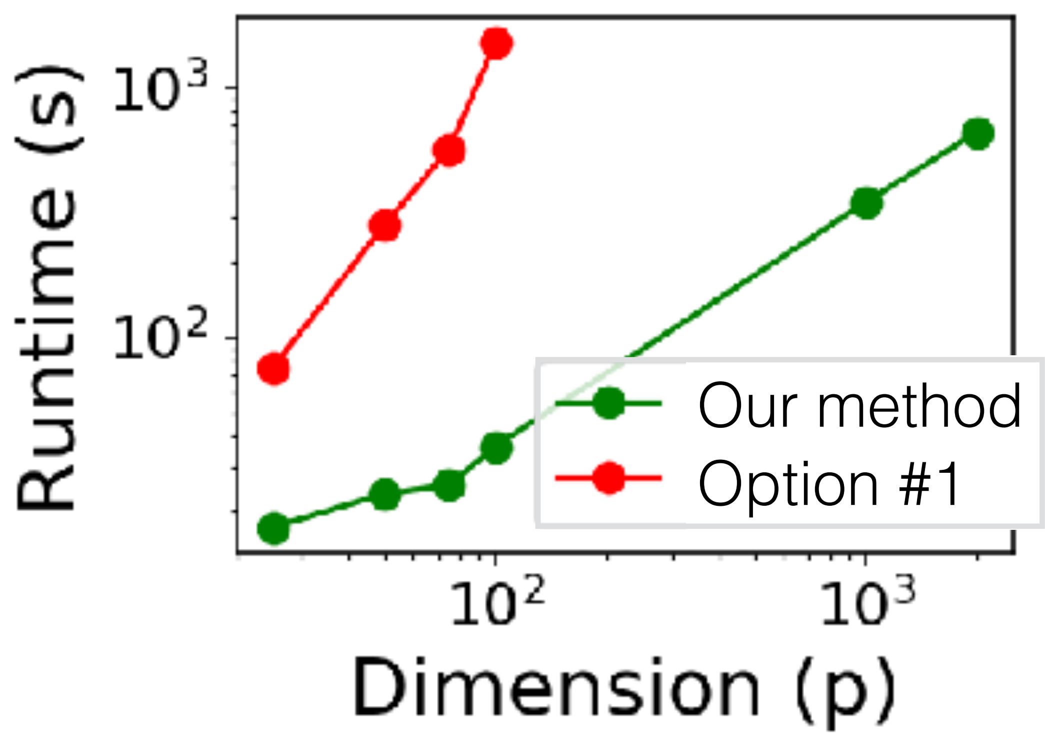
- MCMC option 1: sample θ (p^2 parameters)

Kernel Interaction Sampler vs. Naive MCMC

- MCMC option 1: sample θ (p^2 parameters)
    - Time cost: $O(p^2 N)$

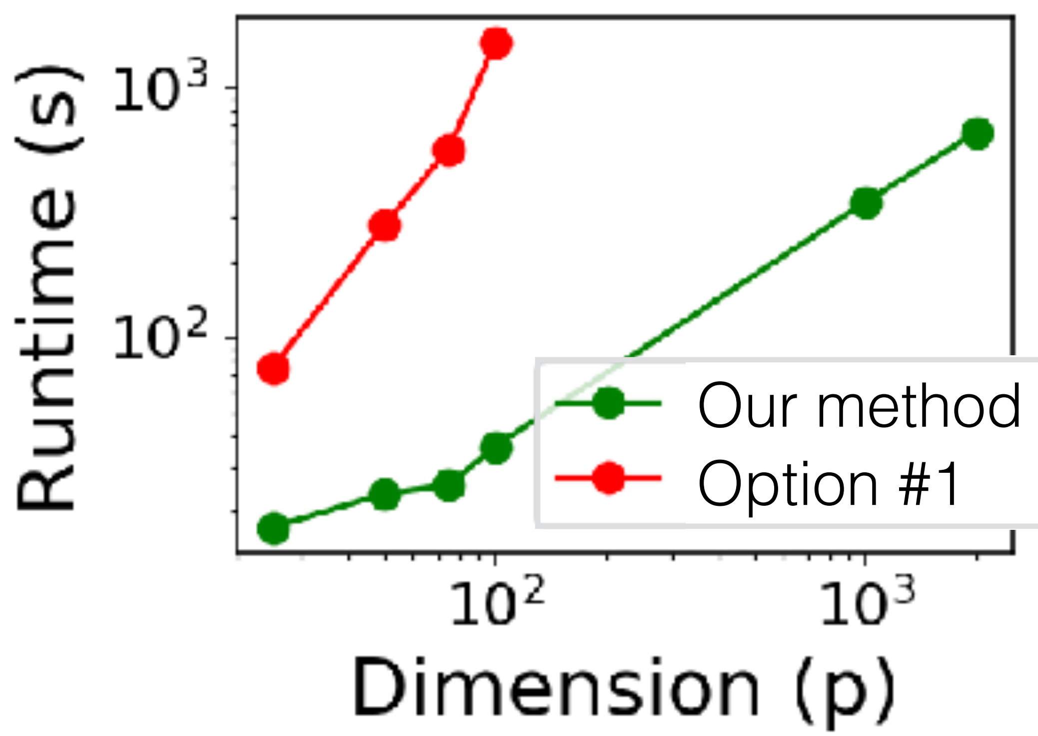
Kernel Interaction Sampler vs. Naive MCMC

- MCMC option 1: sample θ (p^2 parameters)
  - Time cost: $O(p^2 N)$



Kernel Interaction Sampler vs. Naive MCMC

- MCMC option 1: sample θ (p^2 parameters)
  - Time cost: $O(p^2N)$



- Mixing (1000 iters Stan):
  - Option #1: all $\hat{R} > 1.05$
  - Our method: all $\hat{R} < 1.05$

Kernel Interaction Sampler vs. Naive MCMC

Kernel Interaction Sampler vs. Naive MCMC

- MCMC option 2: use conditional conjugacy for θ

Kernel Interaction Sampler vs. Naive MCMC

- MCMC option 2: use conditional conjugacy for θ
  - Compute and invert

$$X^\top X$$

Kernel Interaction Sampler vs. Naive MCMC

- MCMC option 2: use conditional conjugacy for θ
  - Compute and invert

$$X^\top X \quad + \quad \text{prior precision matrix}$$

Kernel Interaction Sampler vs. Naive MCMC

- MCMC option 2: use conditional conjugacy for θ
  - Compute and invert

$$X^\top X$$

Kernel Interaction Sampler vs. Naive MCMC

- MCMC option 2: use conditional conjugacy for θ
  - Compute and invert
    $$\Phi_2(X)^\top \Phi_2(X)$$

Kernel Interaction Sampler vs. Naive MCMC

- MCMC option 2: use conditional conjugacy for θ
  - Compute and invert

$$\Phi_2(X)^\top \Phi_2(X)$$

X: *N*×*p*

Kernel Interaction Sampler vs. Naive MCMC

- MCMC option 2: use conditional conjugacy for θ
  - Compute and invert

$$\Phi_2(X)^\top \Phi_2(X)$$

X: $N \times p$

$\boldsymbol{\Phi}_2$: $N \times p^2$

Kernel Interaction Sampler vs. Naive MCMC

- MCMC option 2: use conditional conjugacy for θ
  - Compute and invert

$$\Phi_2(X)^\top \Phi_2(X)$$

X: $N \times p$

$\boldsymbol{\Phi}_2$: $N \times p^2$

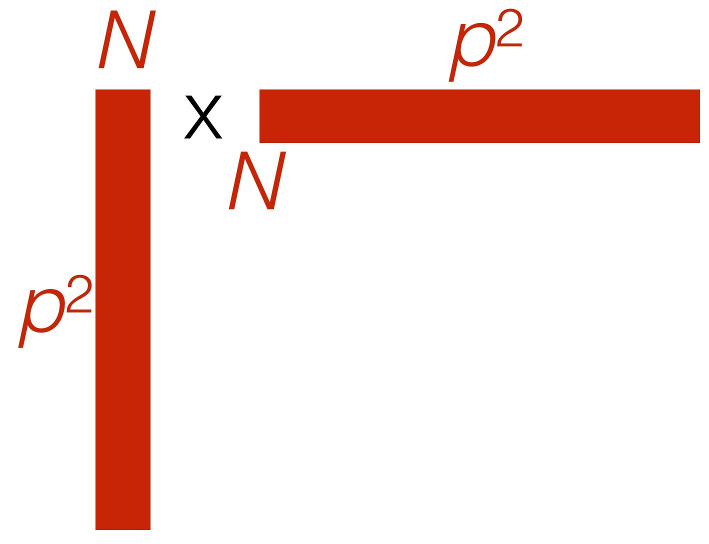
Kernel Interaction Sampler vs. Naive MCMC

- MCMC option 2: use conditional conjugacy for θ
  - Compute and invert

$$\Phi_2(X)^\top \Phi_2(X)$$

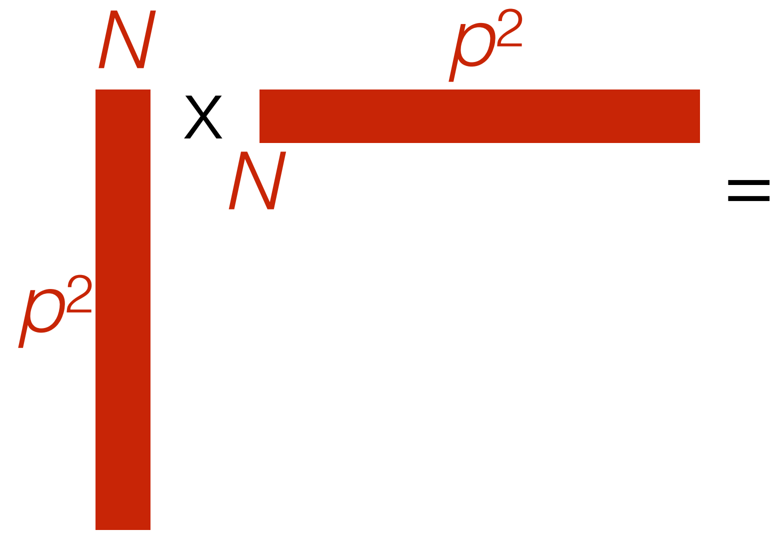
X: $N \times p$

Φ_2: $N \times p^2$



Kernel Interaction Sampler vs. Naive MCMC

- MCMC option 2: use conditional conjugacy for θ
  - Compute and invert
    $$\Phi_2(X)^\top \Phi_2(X)$$
    X: $N \times p$
    Φ_2: $N \times p^2$



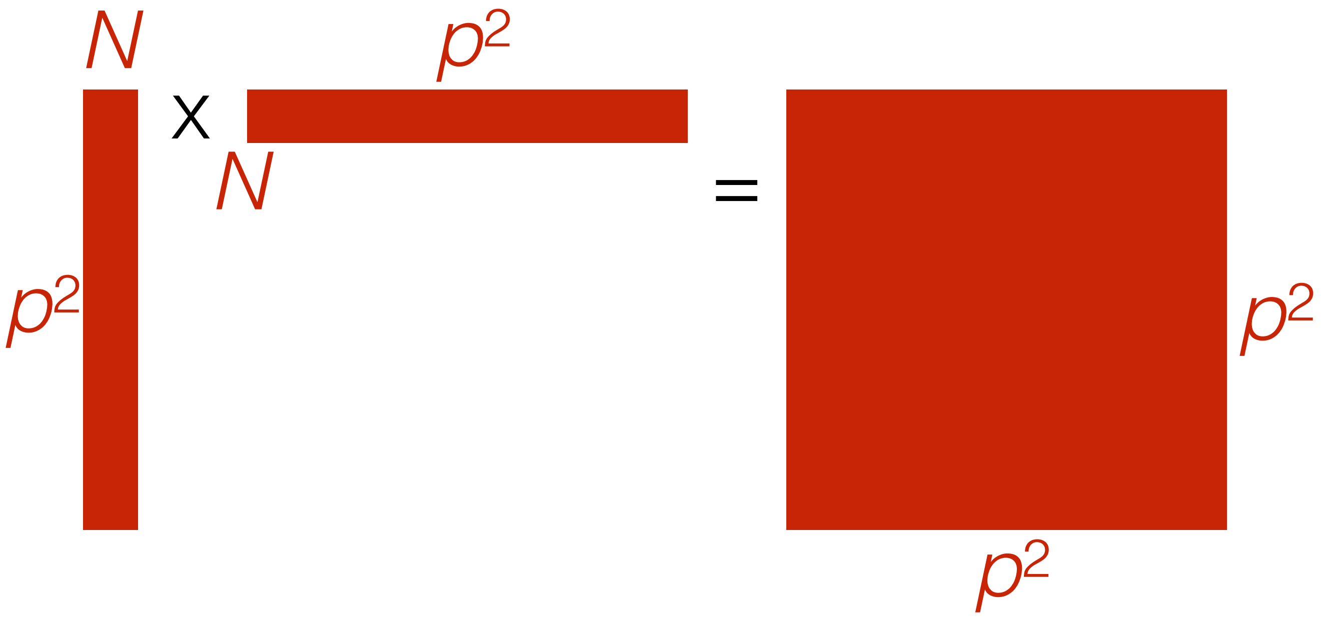
Kernel Interaction Sampler vs. Naive MCMC

- MCMC option 2: use conditional conjugacy for θ
  - Compute and invert

$$\Phi_2(X)^\top \Phi_2(X)$$

X: $N \times p$

$\boldsymbol{\Phi}_2$: $N \times p^2$



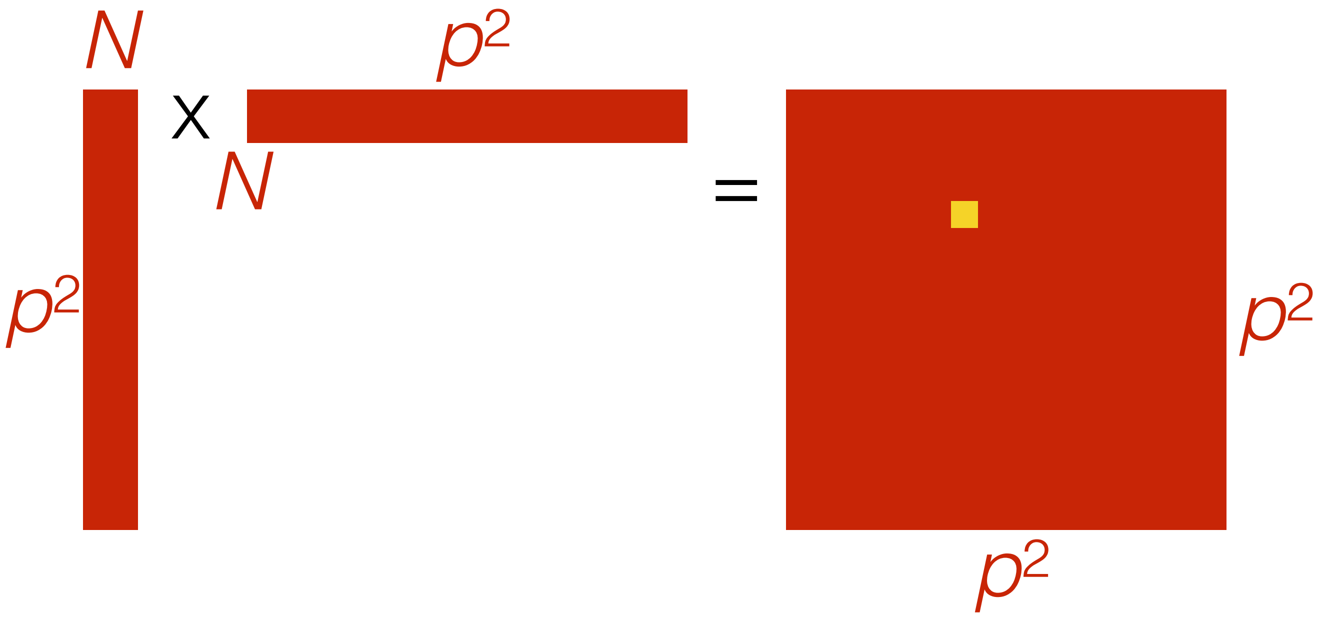
Kernel Interaction Sampler vs. Naive MCMC

- MCMC option 2: use conditional conjugacy for θ
  - Compute and invert

$$\Phi_2(X)^\top \Phi_2(X)$$

X: $N \times p$

$\boldsymbol{\Phi}_2$: $N \times p^2$



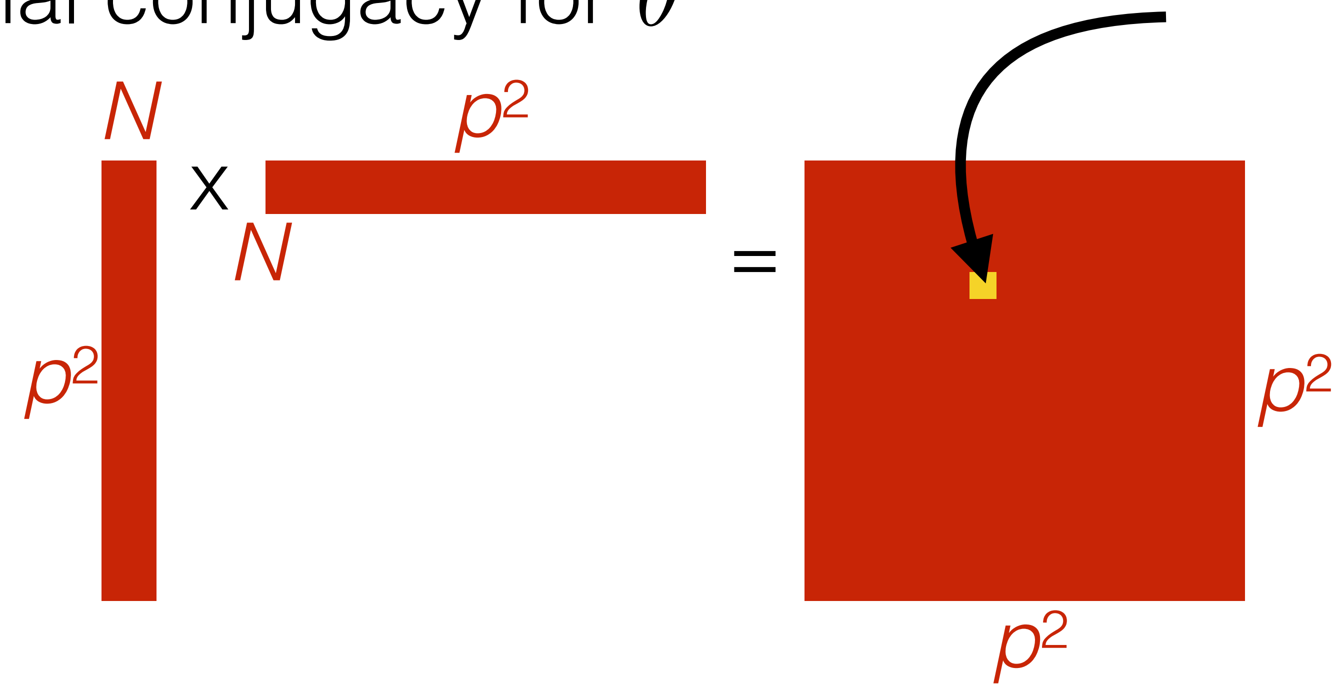
Kernel Interaction Sampler vs. Naive MCMC

- MCMC option 2: use conditional conjugacy for θ
  - Compute and invert

$$\Phi_2(X)^\top \Phi_2(X)$$

X: $N \times p$

Φ_2: $N \times p^2$



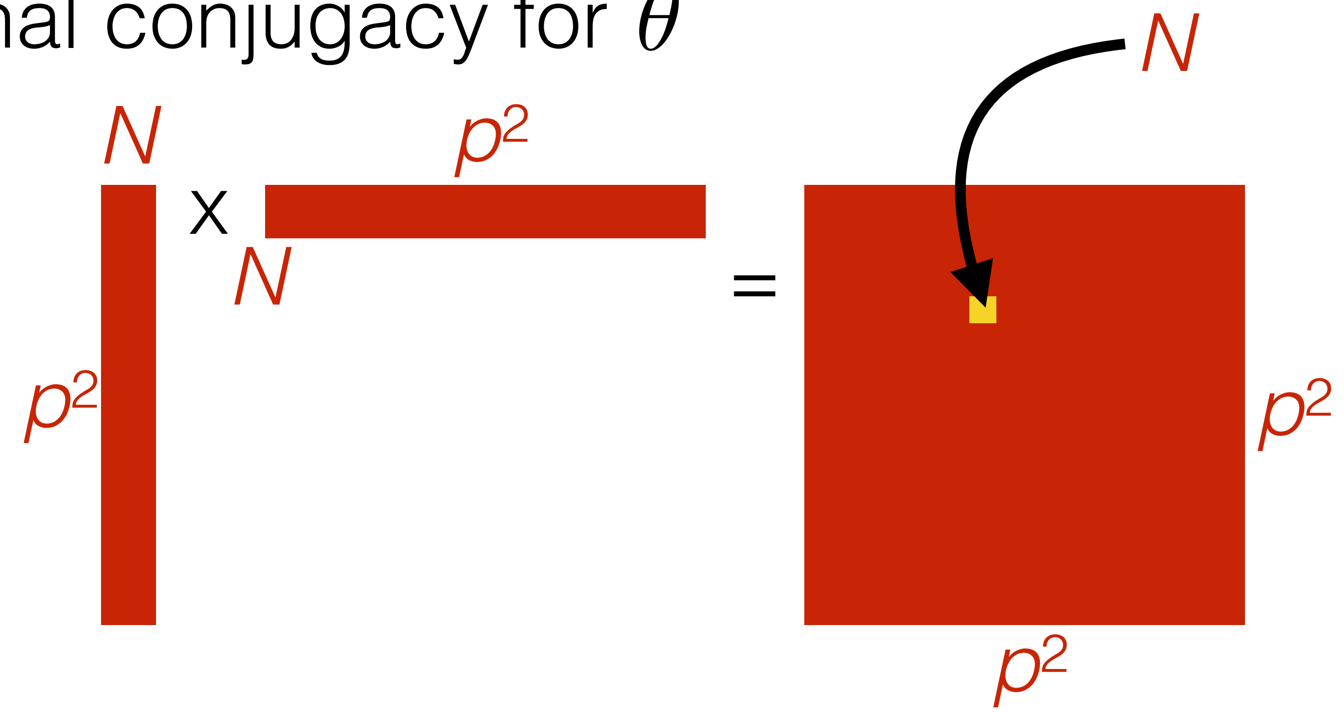
Kernel Interaction Sampler vs. Naive MCMC

- MCMC option 2: use conditional conjugacy for θ
  - Compute and invert

$$\Phi_2(X)^\top \Phi_2(X)$$

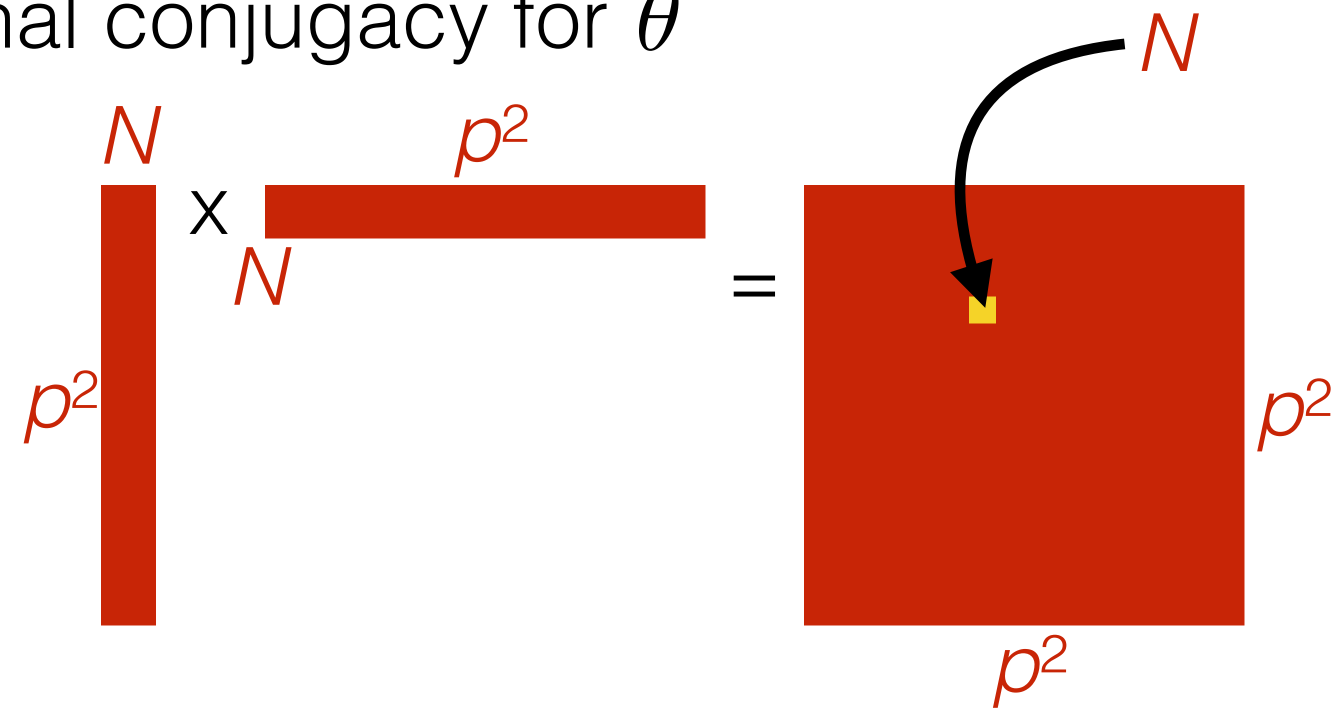
X: *N*×*p*

Φ$_2$: *N*×*p*^{2}



Kernel Interaction Sampler vs. Naive MCMC

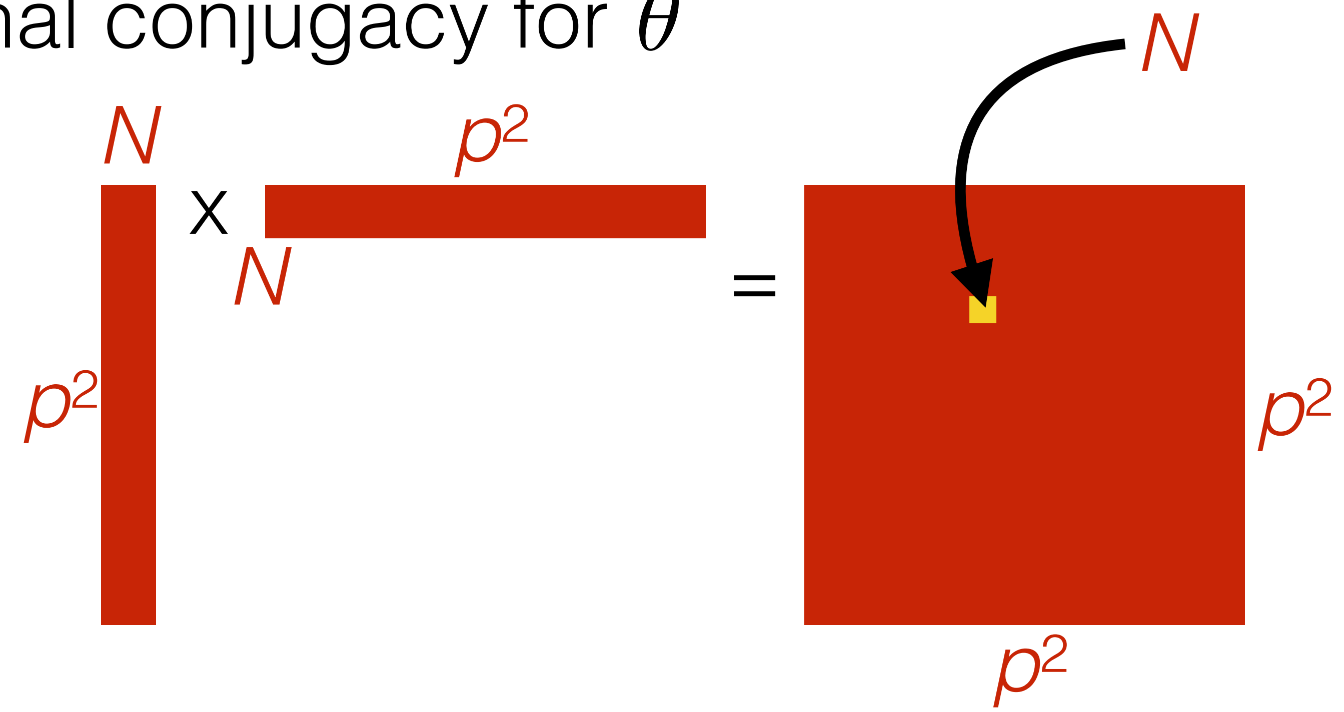
- MCMC option 2: use conditional conjugacy for θ
  - Compute and invert
    $$\Phi_2(X)^\top \Phi_2(X)$$
    X: $N \times p$
    $\boldsymbol{\Phi}_2$: $N \times p^2$



  - Naive time cost: $O(p^4 N + p^6)$

Kernel Interaction Sampler vs. Naive MCMC

- MCMC option 2: use conditional conjugacy for θ
  - Compute and invert
    $$\Phi_2(X)^\top \Phi_2(X)$$
    X: $N \times p$
    $\boldsymbol{\Phi}_2$: $N \times p^2$



  - Naive time cost: $O(p^4 N + p^6)$
  - Woodbury time cost: $O(p^2 N^2 + N^3)$

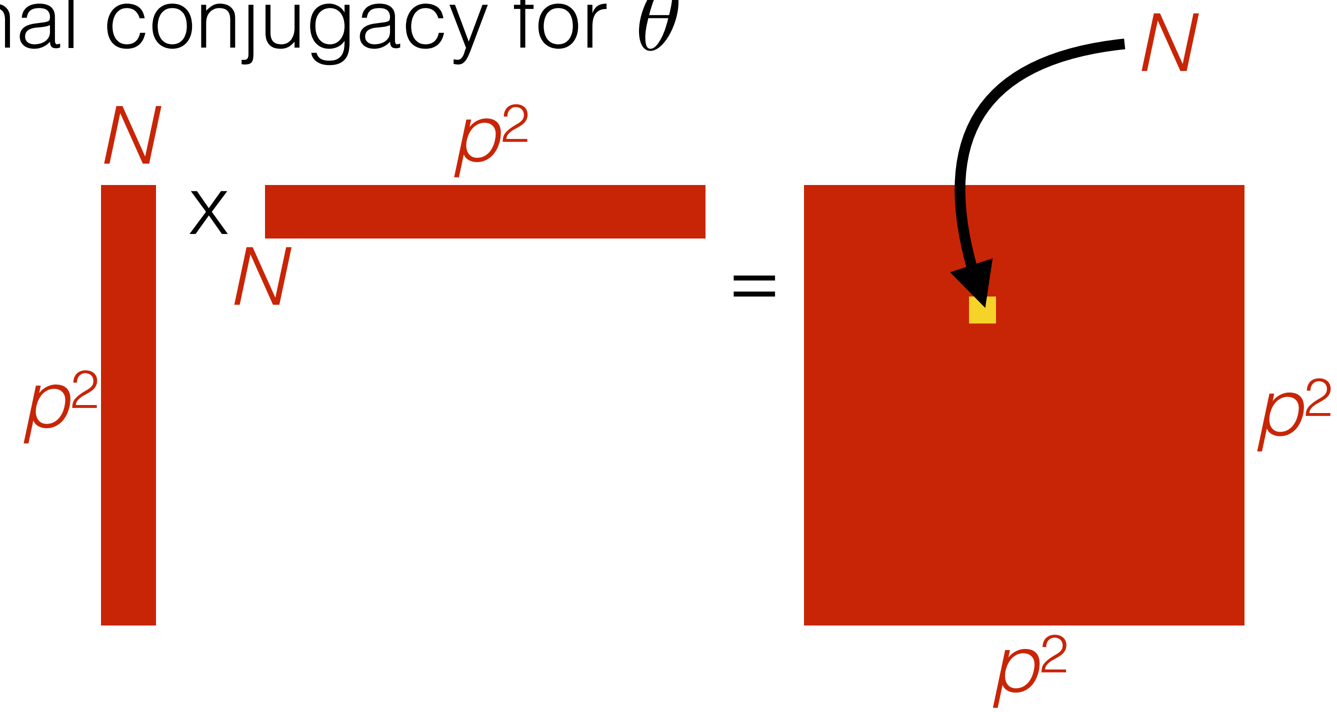
Kernel Interaction Sampler vs. Naive MCMC

- MCMC option 2: use conditional conjugacy for θ
  - Compute and invert

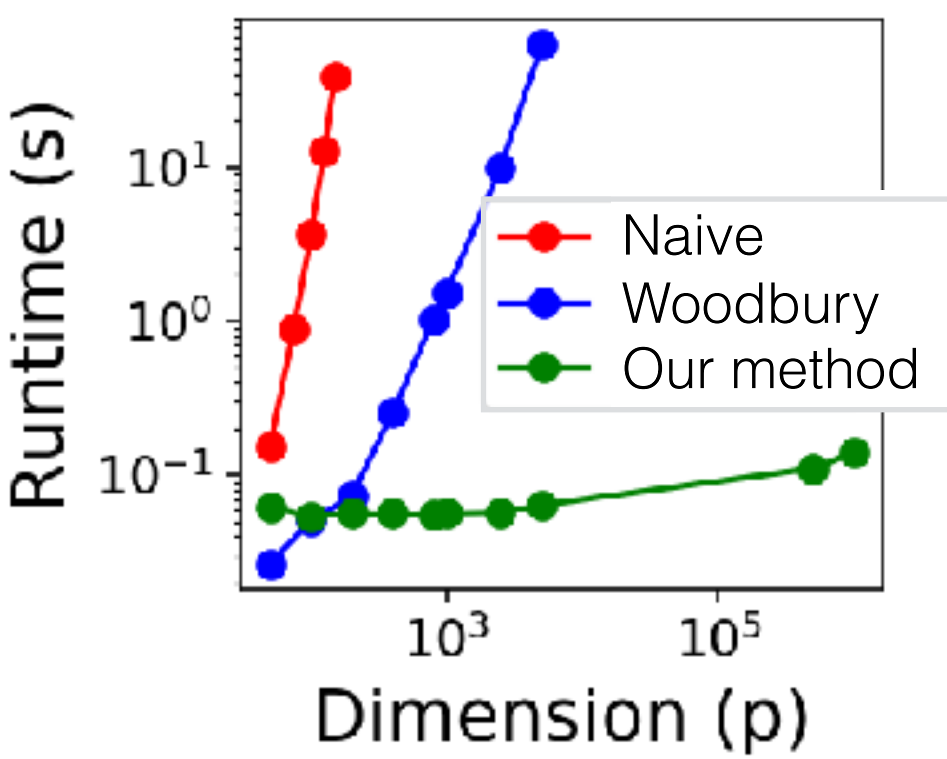
$$\Phi_2(X)^\top \Phi_2(X)$$

X: $N \times p$

$\boldsymbol{\Phi}_2$: $N \times p^2$



- Naive time cost: $O(p^4 N + p^6)$
- Woodbury time cost: $O(p^2 N^2 + N^3)$



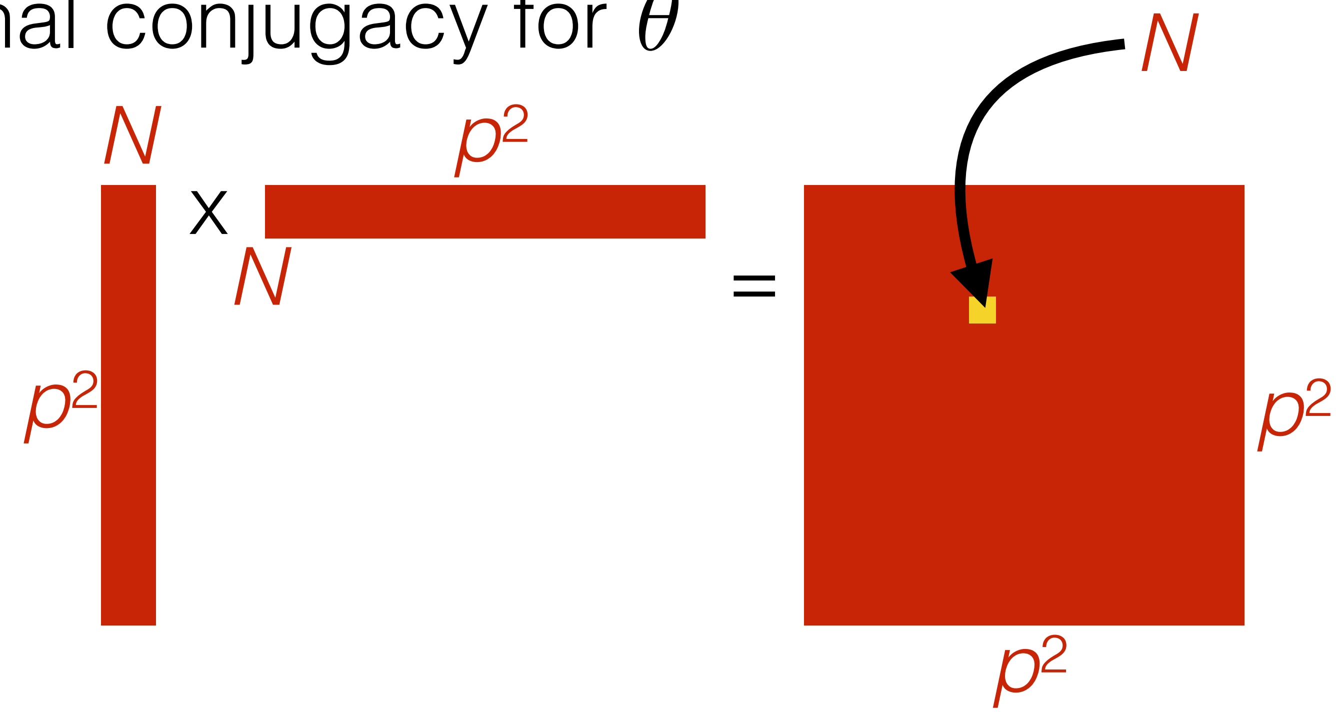
Kernel Interaction Sampler vs. Naive MCMC

- MCMC option 2: use conditional conjugacy for θ
  - Compute and invert

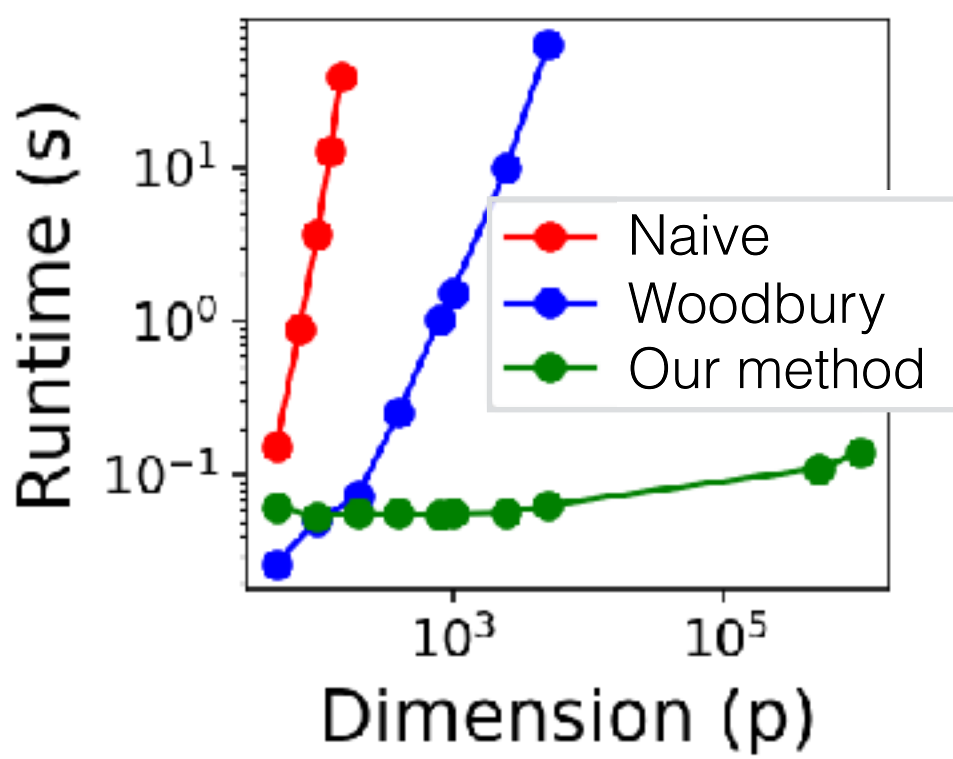
$$\Phi_2(X)^\top \Phi_2(X)$$

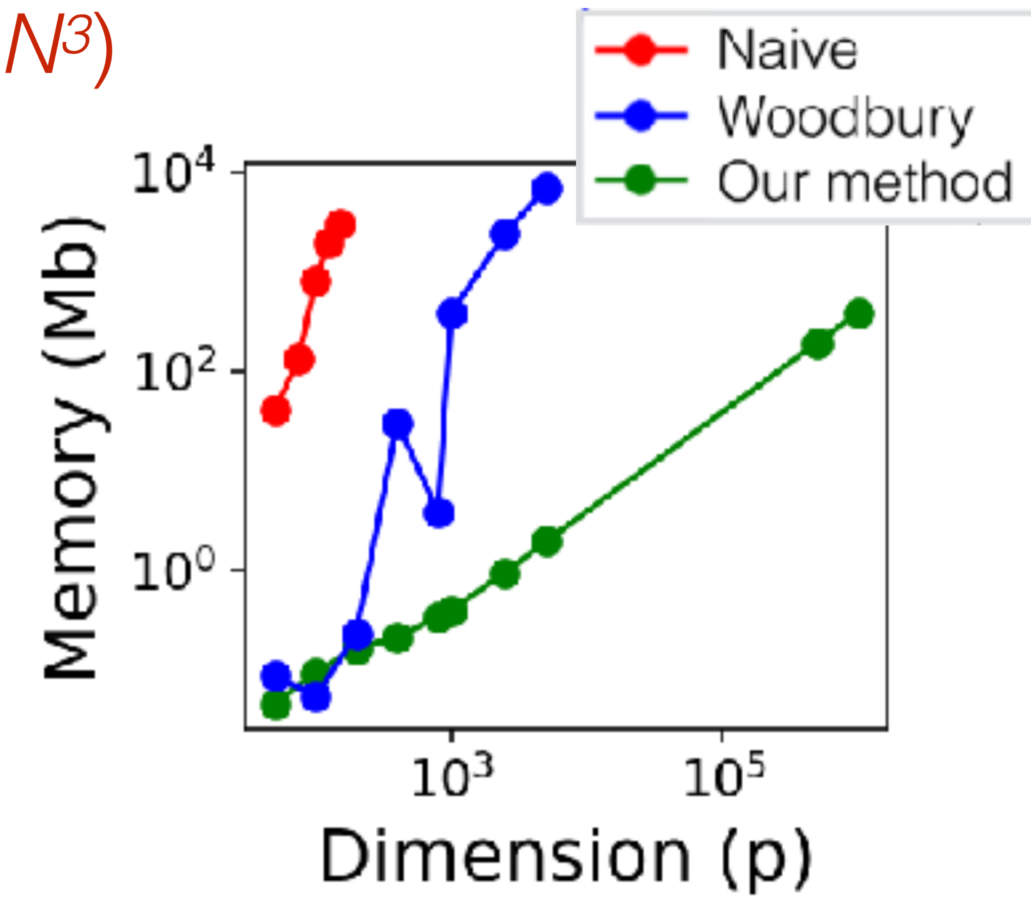
X: $N \times p$

$\boldsymbol{\Phi}_2$: $N \times p^2$



- Naive time cost: $O(p^4 N + p^6)$
- Woodbury time cost: $O(p^2 N^2 + N^3)$





Kernel Interaction Sampler vs. Naive MCMC

- Compute and invert

$$\Phi_2(X)^\top \Phi_2(X)$$

X: $N \times p$

$\boldsymbol{\Phi}_2$: $N \times p^2$

Kernel Interaction Sampler vs. Naive MCMC

- Compute and invert

$$\Phi_2(X)^\top \Phi_2(X)$$

X: $N \times p$

$\boldsymbol{\Phi}_2$: $N \times p^2$

Kernel Interaction Sampler vs. Naive MCMC

use conditional conjugacy for $\theta^T \Phi_2(X)$

- Compute and invert

$$\Phi_2(X)^\top \Phi_2(X)$$

X: $N \times p$

$\boldsymbol{\Phi}_2$: $N \times p^2$

Kernel Interaction Sampler vs. Naive MCMC

- Our approach: use conditional conjugacy for $\theta^T \Phi_2(X)$
  - Compute and invert

$$\Phi_2(X)^\top \Phi_2(X)$$

X: *N*×*p*

$\boldsymbol{\Phi}_2$: *N*×p^2

Kernel Interaction Sampler vs. Naive MCMC

- Our approach: use conditional conjugacy for $\theta^T \Phi_2(X)$
  - Compute and invert

$$\Phi_2(X)^\top \Phi_2(X)$$

X: $N \times p$

$\boldsymbol{\Phi}_2$: $N \times p^2$

Kernel Interaction Sampler vs. Naive MCMC

- Our approach: use conditional conjugacy for $\theta^T \Phi_2(X)$
  - Compute and invert

$$\Phi_2(X)\Phi_2(X)^\top \Phi_2(X)\Phi_2(X)^\top$$

X: $N \times p$

$\boldsymbol{\Phi}_2$: $N \times p^2$

Kernel Interaction Sampler vs. Naive MCMC

- Our approach: use conditional conjugacy for $\theta^T \Phi_2(X)$
  - Compute and invert

$$\Phi_2(X)\Phi_2(X)^\top\Phi_2(X)\Phi_2(X)^\top$$

X: $N \times p$

$\boldsymbol{\Phi}_2$: $N \times p^2$

Kernel Interaction Sampler vs. Naive MCMC

- Our approach: use conditional conjugacy for $\theta^T \Phi_2(X)$
    - Compute and invert

$$\Phi_2(X)\Phi_2(X)^\top \Phi_2(X)\Phi_2(X)^\top$$

X: $N \times p$

$\boldsymbol{\Phi}_2$: $N \times p^2$

Kernel Interaction Sampler vs. Naive MCMC

- Our approach: use conditional conjugacy for $\theta^T \Phi_2(X)$
  - Compute and invert

$$\Phi_2(X)\Phi_2(X)^\top$$

X: $N \times p$

$\boldsymbol{\Phi}_2$: $N \times p^2$

Kernel Interaction Sampler vs. Naive MCMC

- Our approach: use conditional conjugacy for $\theta^T \Phi_2(X)$
  - Compute and invert

$$\Phi_2(X)\Phi_2(X)^\top$$

X: $N \times p$

$\boldsymbol{\Phi}_2$: $N \times p^2$

Kernel Interaction Sampler vs. Naive MCMC

- Our approach: use conditional conjugacy for $\theta^T \Phi_2(X)$
  - Compute and invert

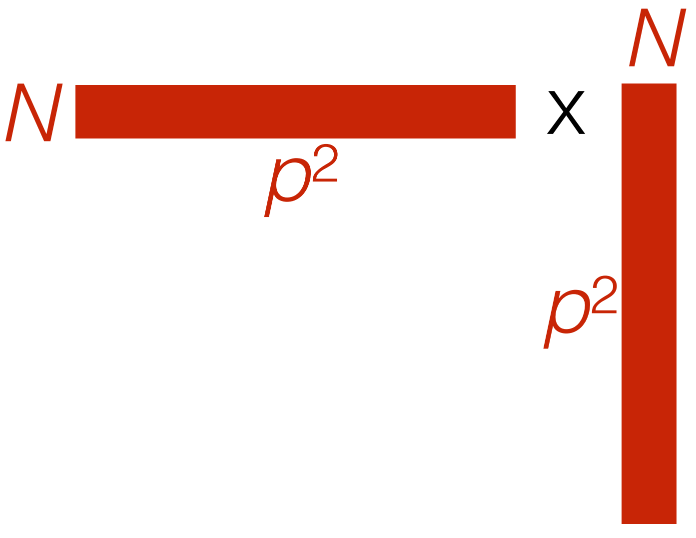
$$\Phi_2(X)\Phi_2(X)^\top$$

X: $N \times p$

$\boldsymbol{\Phi}_2$: $N \times p^2$

Kernel Interaction Sampler vs. Naive MCMC

- Our approach: use conditional conjugacy for $\theta^T \Phi_2(X)$
  - Compute and invert
    $$\Phi_2(X)\Phi_2(X)^\top$$
    X: $N \times p$
    $\boldsymbol{\Phi}_2$: $N \times p^2$



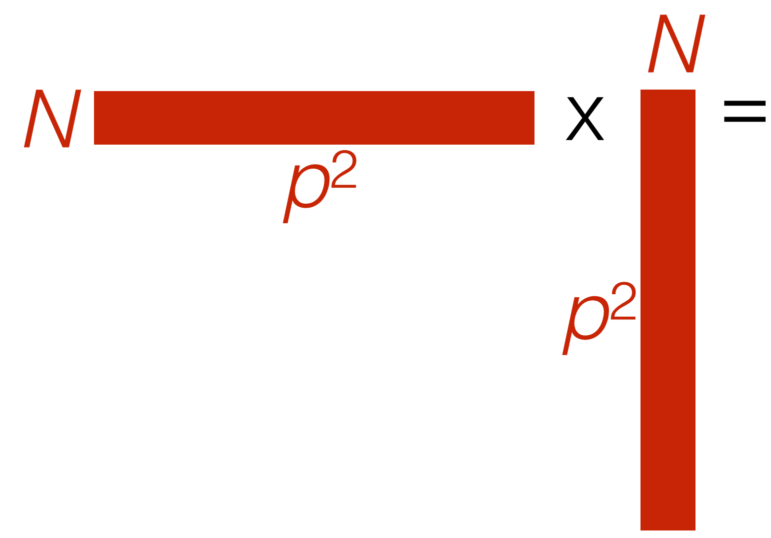
Kernel Interaction Sampler vs. Naive MCMC

- Our approach: use conditional conjugacy for $\theta^T \Phi_2(X)$
  - Compute and invert

$$\Phi_2(X)\Phi_2(X)^\top$$

X: $N \times p$

$\boldsymbol{\Phi}_2$: $N \times p^2$



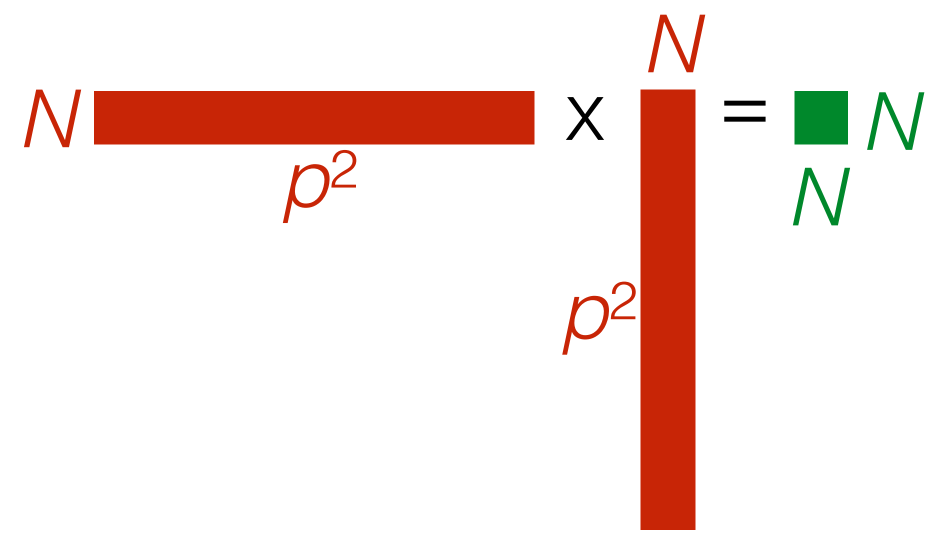
Kernel Interaction Sampler vs. Naive MCMC

- Our approach: use conditional conjugacy for $\theta^T \Phi_2(X)$
  - Compute and invert

$$\Phi_2(X)\Phi_2(X)^\top$$

X: $N \times p$

$\boldsymbol{\Phi}_2$: $N \times p^2$



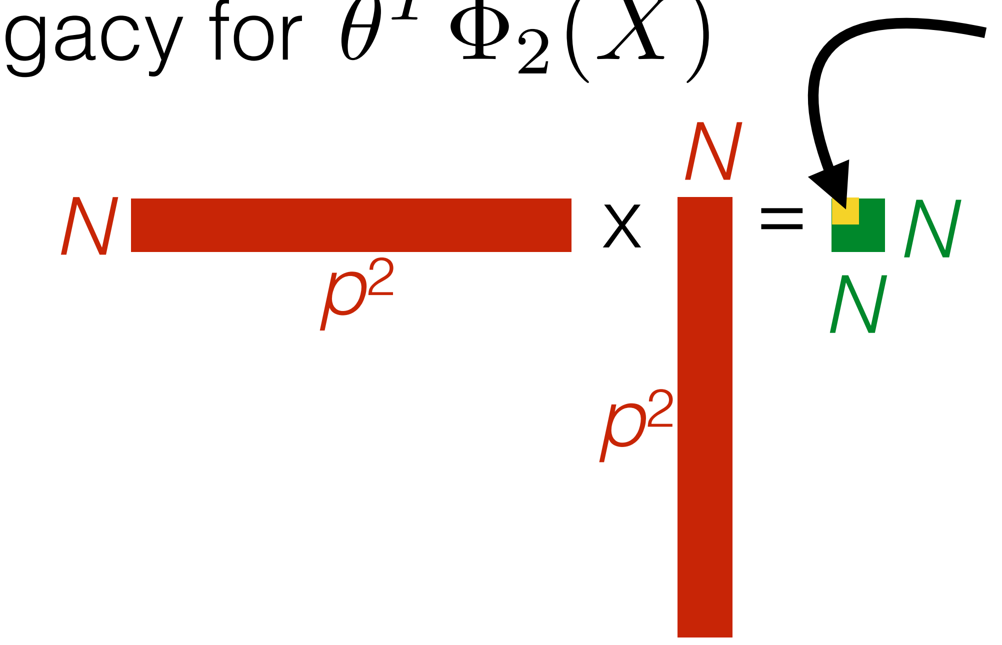
Kernel Interaction Sampler vs. Naive MCMC

- Our approach: use conditional conjugacy for $\theta^T \Phi_2(X)$
  - Compute and invert

$$\Phi_2(X)\Phi_2(X)^\top$$

X: $N \times p$

$\boldsymbol{\Phi}_2$: $N \times p^2$



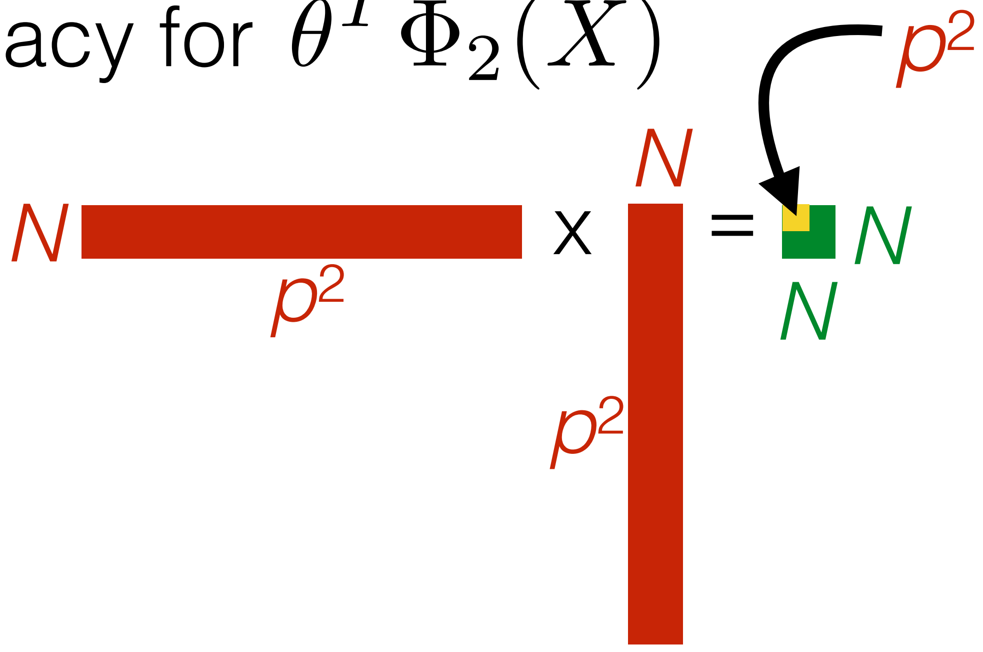
Kernel Interaction Sampler vs. Naive MCMC

- Our approach: use conditional conjugacy for $\theta^T \Phi_2(X)$
  - Compute and invert

$$\Phi_2(X)\Phi_2(X)^\top$$

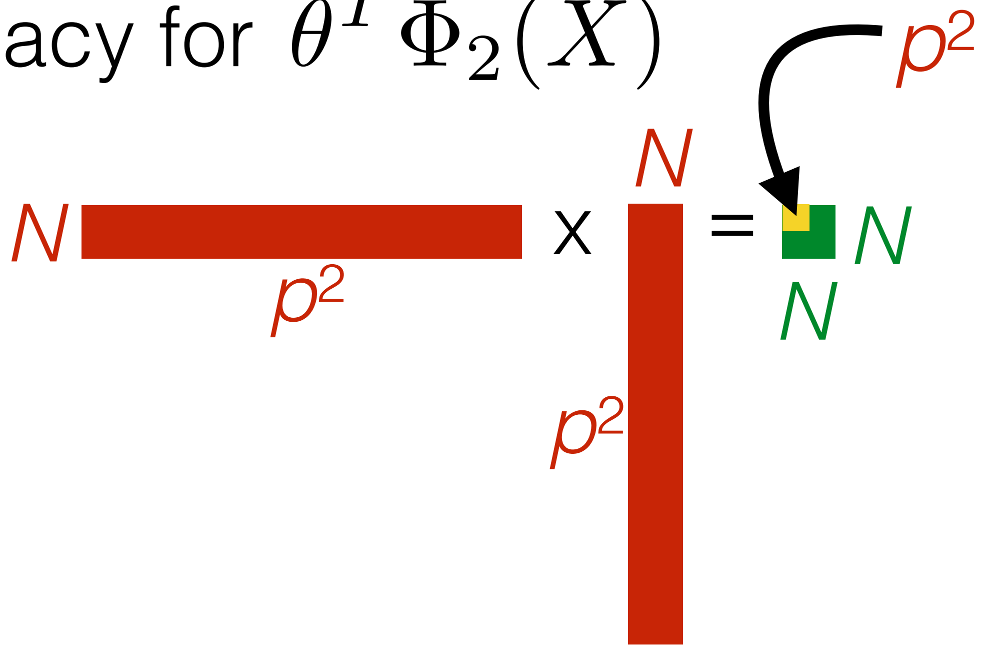
X: *N*×*p*

$\boldsymbol{\Phi}_2$: *N*×p^2



Kernel Interaction Sampler vs. Naive MCMC

- Our approach: use conditional conjugacy for $\theta^T \Phi_2(X)$
  - Compute and invert
    $$\Phi_2(X)\Phi_2(X)^\top$$
    X: $N \times p$
    $\boldsymbol{\Phi}_2$: $N \times p^2$
  - Kernel trick: $O(p)$ cost



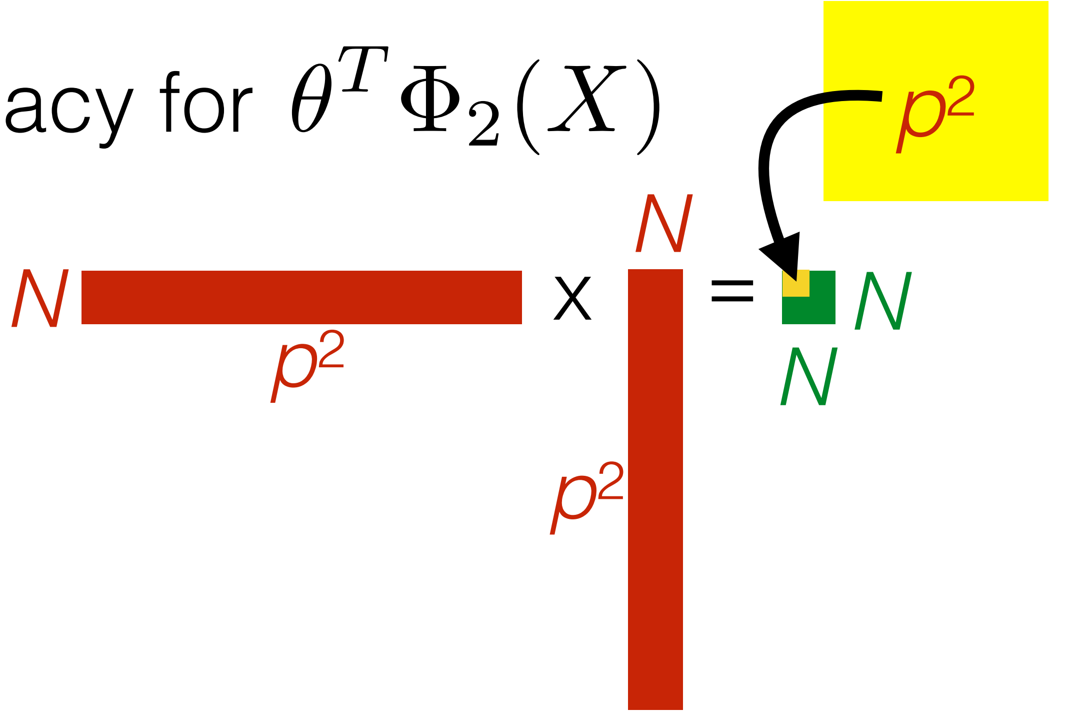
Kernel Interaction Sampler vs. Naive MCMC

- Our approach: use conditional conjugacy for $\theta^T \Phi_2(X)$
  - Compute and invert

    $$\Phi_2(X)\Phi_2(X)^\top$$

    X: $N \times p$

    $\boldsymbol{\Phi}_2$: $N \times p^2$
  - Kernel trick: $O(p)$ cost



Kernel Interaction Sampler vs. Naive MCMC

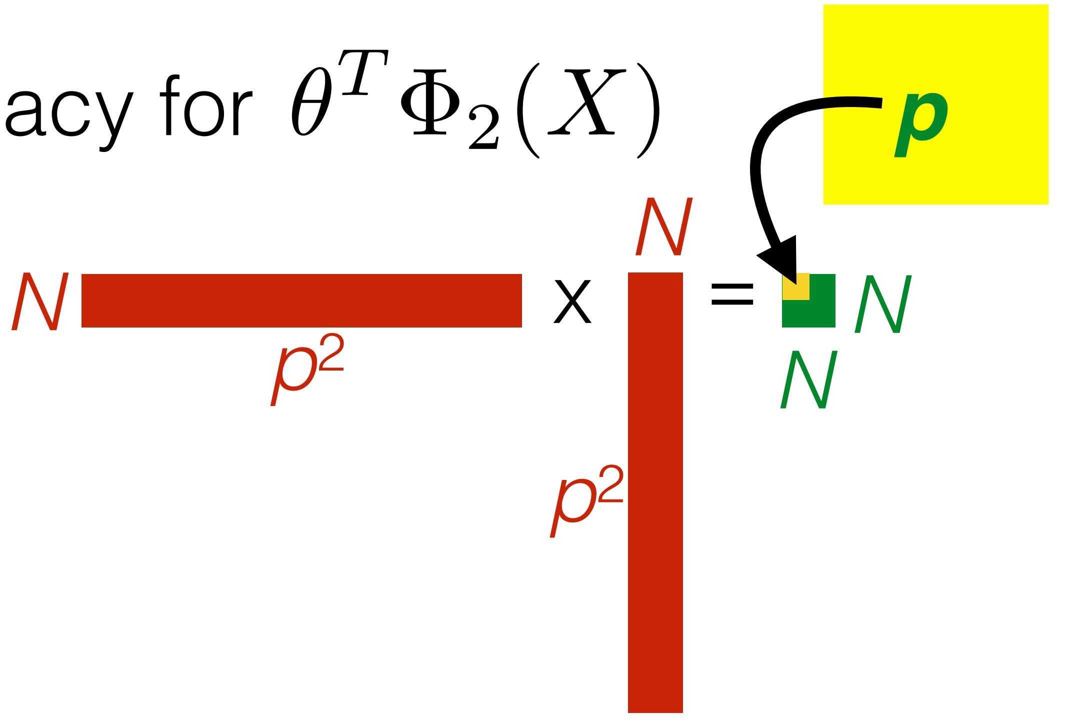
- Our approach: use conditional conjugacy for $\theta^T \Phi_2(X)$
  - Compute and invert

    $$\Phi_2(X)\Phi_2(X)^\top$$

    X: $N \times p$

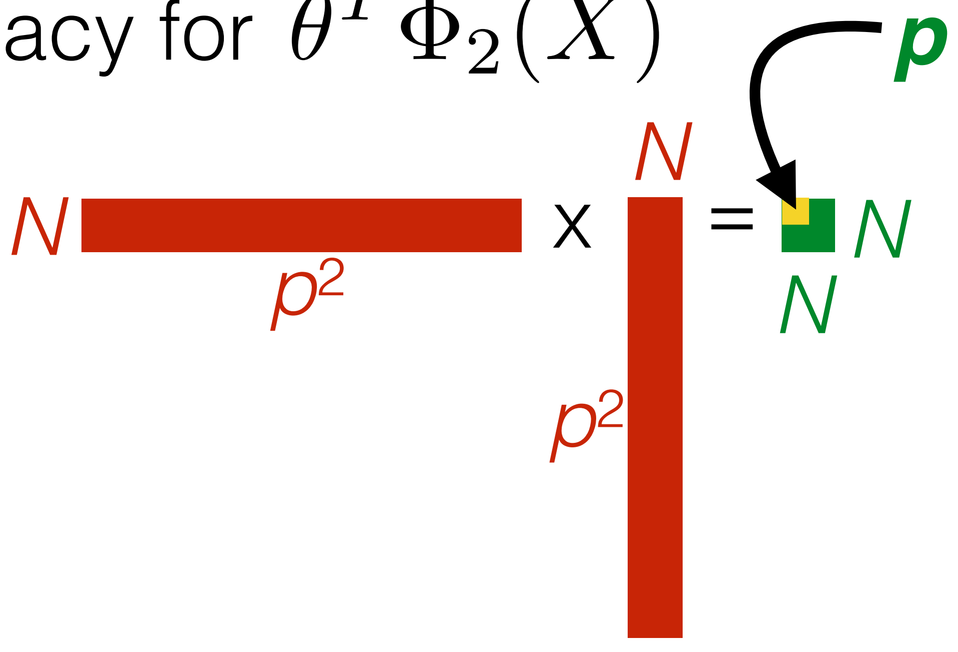
    $\boldsymbol{\Phi}_2$: $N \times p^2$

  - Kernel trick: $O(p)$ cost



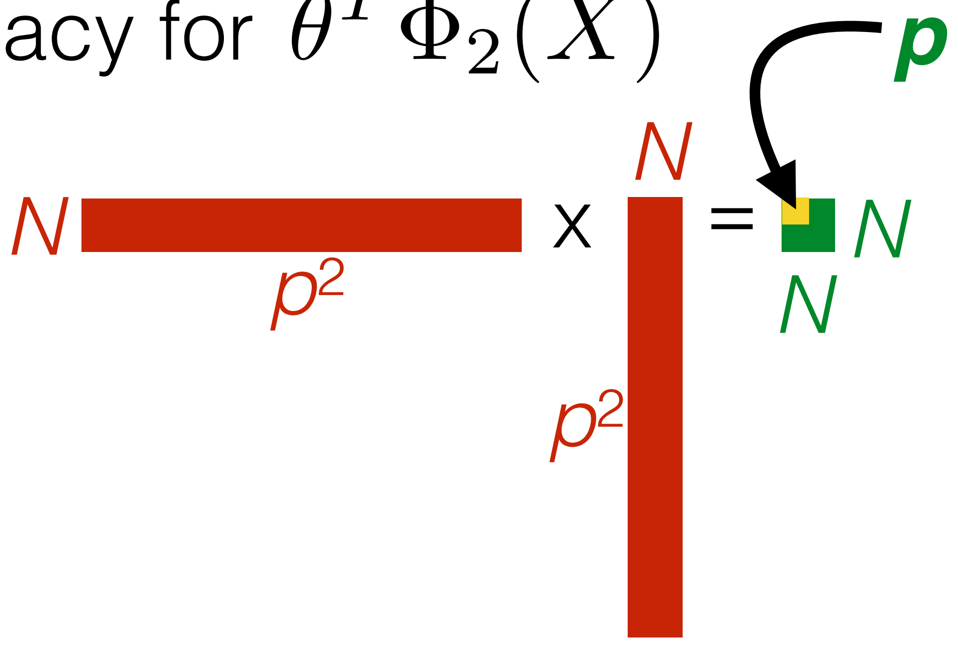
Kernel Interaction Sampler vs. Naive MCMC

- Our approach: use conditional conjugacy for $\theta^T \Phi_2(X)$
  - Compute and invert
    $$\Phi_2(X)\Phi_2(X)^\top$$
    X: $N \times p$
    Φ_2: $N \times p^2$
  - Kernel trick: $O(p)$ cost



Kernel Interaction Sampler vs. Naive MCMC

- Our approach: use conditional conjugacy for $\theta^T \Phi_2(X)$
  - Compute and invert
    $$\Phi_2(X)\Phi_2(X)^\top$$
    X: $N \times p$
    $\boldsymbol{\Phi}_2$: $N \times p^2$
  - Kernel trick: $O(p)$ cost
  - Our time cost: $O(\boldsymbol{p}N^2 + N^3)$



Reporting: Kernel Interaction Trick

Reporting: Kernel Interaction Trick

- Can access posterior of $g = \theta^T \Phi_2$ in *O*(*p*) time per iteration

Reporting: Kernel Interaction Trick

- Can access posterior of $g = \theta^T \Phi_2$ in *O*(*p*) time per iteration
- *But* our goal is to find main and interaction effects

Reporting: Kernel Interaction Trick

- Can access posterior of $g = \theta^T \Phi_2$ in *O*(*p*) time per iteration
- *But* our goal is to find main and interaction effects
- Step A: Report posterior of θ_{x_i} or $\theta_{x_i x_j}$ in *O*(1) time

Reporting: Kernel Interaction Trick

- Can access posterior of $g = \theta^T \Phi_2$ in *O*(*p*) time per iteration
- *But* our goal is to find main and interaction effects
- Step A: Report posterior of θ_{x_i} or $\theta_{x_i x_j}$ in *O*(1) time

$$e_i = [0, 0, \ldots, 0, 1, 0, \ldots, 0]$$

i^{th} position

Reporting: Kernel Interaction Trick

- Can access posterior of $g = \theta^T \Phi_2$ in *O*(*p*) time per iteration
- *But* our goal is to find main and interaction effects
- Step A: Report posterior of θ_{x_i} or $\theta_{x_i x_j}$ in *O*(1) time

$$e_i = [0, 0, \ldots, 0, 1, 0, \ldots, 0]$$

i^{th} position

$$g(e_i) = \theta_{x_i} + \theta_{x_i^2}$$

Reporting: Kernel Interaction Trick

- Can access posterior of $g = \theta^T \Phi_2$ in *O*(*p*) time per iteration
- *But* our goal is to find main and interaction effects
- Step A: Report posterior of θ_{x_i} or $\theta_{x_i x_j}$ in *O*(1) time

$$e_i = [0, 0, \ldots, 0, 1, 0, \ldots, 0]$$

i^{th} position

$$g(e_i) = \theta_{x_i} + \theta_{x_i^2}$$

$$g(-e_i) = -\theta_{x_i} + \theta_{x_i^2}$$

Reporting: Kernel Interaction Trick

- Can access posterior of $g = \theta^T \Phi_2$ in *O*(*p*) time per iteration
- *But* our goal is to find main and interaction effects
- Step A: Report posterior of θ_{x_i} or $\theta_{x_i x_j}$ in *O*(1) time

$$e_i = [0, 0, \ldots, 0, 1, 0, \ldots, 0]$$

i^{th} position

$$g(e_i) = \theta_{x_i} + \theta_{x_i^2}$$

$$g(-e_i) = -\theta_{x_i} + \theta_{x_i^2}$$

$$\frac{g(e_i) - g(-e_i)}{2} = \theta_{x_i}$$

Reporting: Kernel Interaction Trick

- Can access posterior of $g = \theta^T \Phi_2$ in *O*(*p*) time per iteration
- *But* our goal is to find main and interaction effects
- Step A: Report posterior of θ_{x_i} or $\theta_{x_i x_j}$ in *O*(1) time

$$e_i = [0, 0, \ldots, 0, 1, 0, \ldots, 0]$$

i^{th} position

$$g(e_i) = \theta_{x_i} + \theta_{x_i^2}$$

$$g(-e_i) = -\theta_{x_i} + \theta_{x_i^2}$$

$$\frac{g(e_i) - g(-e_i)}{2} = \theta_{x_i}$$

- Step B: Find $k \ll p$ sparse main effects: takes *O*(*p*) time

Reporting: Kernel Interaction Trick

- Can access posterior of $g = \theta^T \Phi_2$ in *O*(*p*) time per iteration
- *But* our goal is to find main and interaction effects
- Step A: Report posterior of θ_{x_i} or $\theta_{x_i x_j}$ in *O*(1) time

$$e_i = [0, 0, \ldots, 0, 1, 0, \ldots, 0]$$

i^{th} position

$$g(e_i) = \theta_{x_i} + \theta_{x_i^2}$$

$$g(-e_i) = -\theta_{x_i} + \theta_{x_i^2}$$

$$\frac{g(e_i) - g(-e_i)}{2} = \theta_{x_i}$$

- Step B: Find $k \ll p$ sparse main effects: takes *O*(*p*) time
- Step C: Report just the *k*^{2} strong-hierarchy interaction effects: takes *O*(*k*^{2}) time

Roadmap

- Setup: Discovering main and interaction effects
- Our method
  - A Bayesian generative model
  - Fast inference
  - Fast reporting of results
- Experiments on simulated and real data

Roadmap

- Setup: Discovering main and interaction effects
- Our method
    - A Bayesian generative model
    - Fast inference
    - Fast reporting of results
- Experiments on simulated and real data

Timing vs. LASSO-based methods

Timing vs. LASSO-based methods

- LASSO (pairs, hierarchical): $O(p^2)$ per iteration

Timing vs. LASSO-based methods

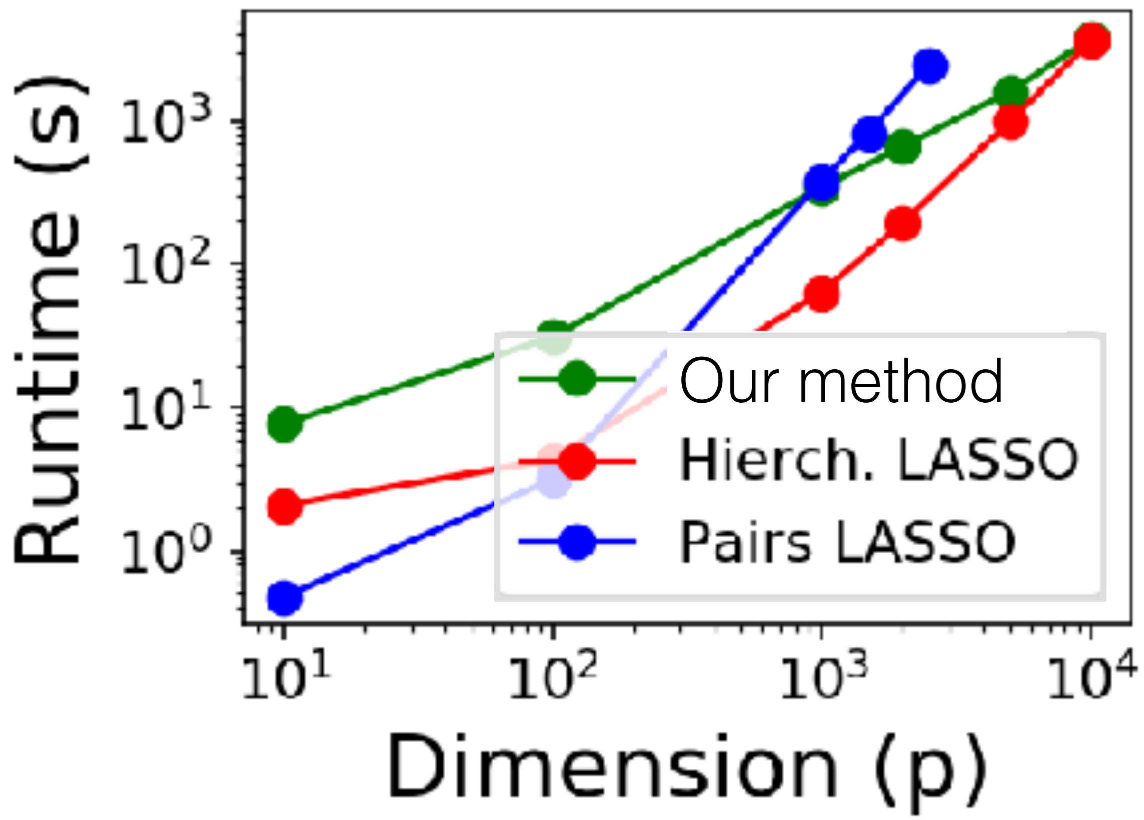
- LASSO (pairs, hierarchical): $O(p^2)$ per iteration [Lim, Hastie 2015]

Timing vs. LASSO-based methods

- LASSO (pairs, hierarchical): $O(p^2)$ per iteration [Lim, Hastie 2015]
- Our method: $O(p)$ per iteration

Timing vs. LASSO-based methods

- LASSO (pairs, hierarchical): $O(p^2)$ per iteration [Lim, Hastie 2015]
- Our method: $O(p)$ per iteration
- Competitive empirically for moderate p:



Experiments: Simulated

Experiments: Simulated

- 36 different simulated data sets (so know true effects)

Experiments: Simulated

- 36 different simulated data sets (so know true effects)
  - Up to $p = 500$ ➔ ≈125,000 total parameters

Experiments: Simulated, Selection

- 36 different simulated data sets (so know true effects)
  - Up to $p = 500$ ➔ ≈125,000 total parameters

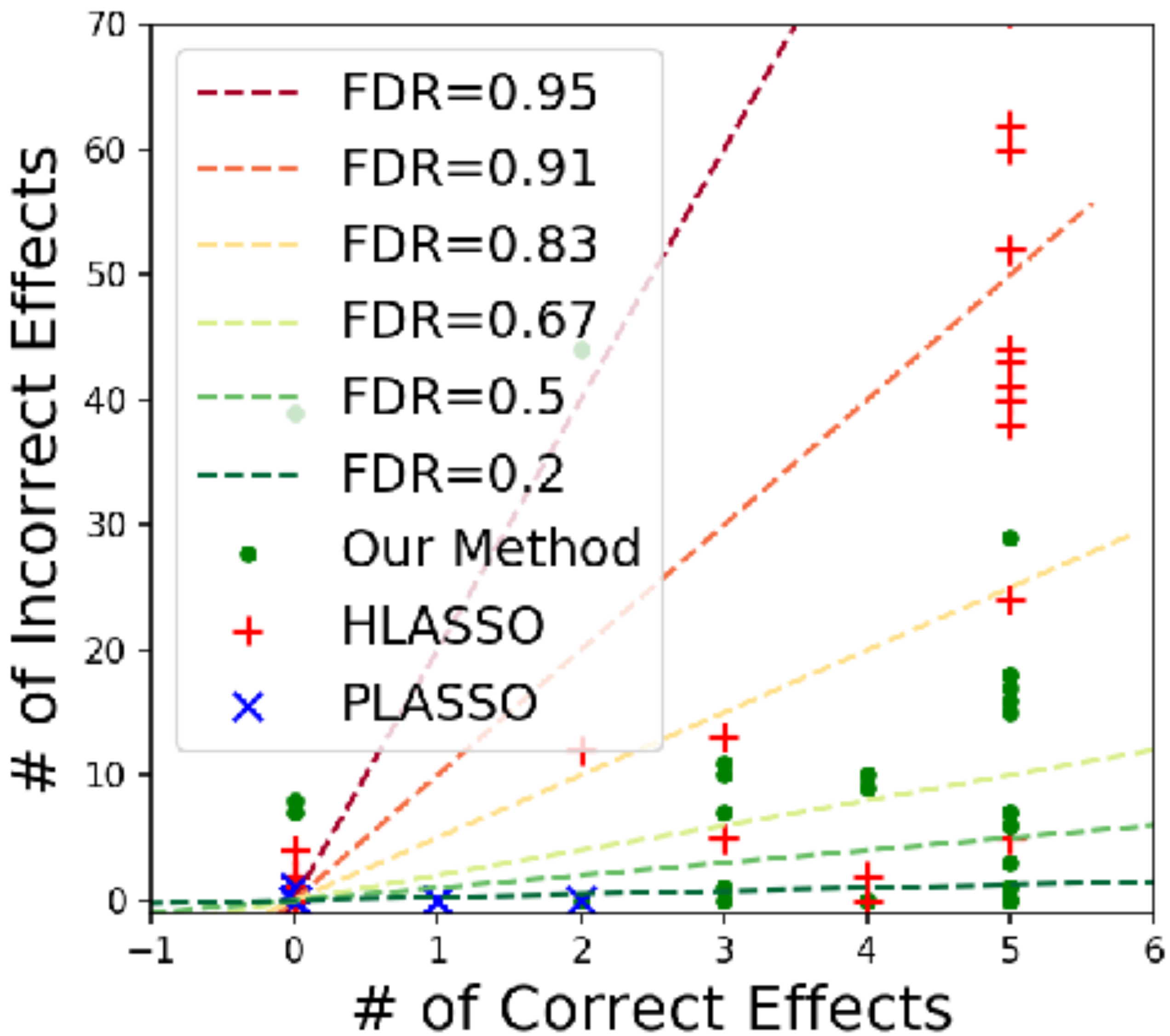
Experiments: Simulated, Selection

- 36 different simulated data sets (so know true effects)
  - Up to $p = 500$ ➔ ≈125,000 total parameters
- False discovery rate (FDR): proportion incorrect

Experiments: Simulated, Selection

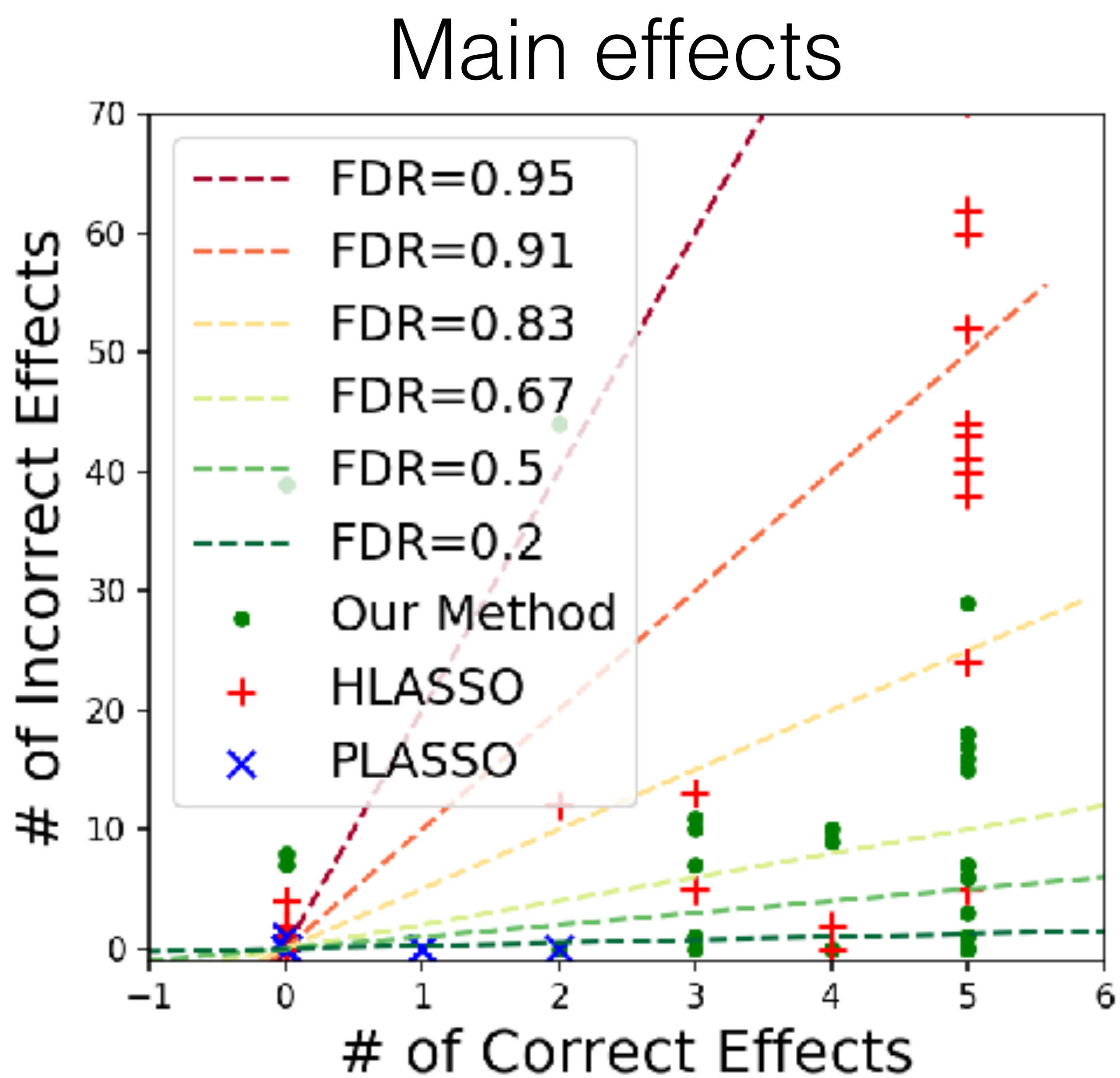
- 36 different simulated data sets (so know true effects)
  - Up to p = 500 ➔ ≈125,000 total parameters
- False discovery rate (FDR): proportion incorrect

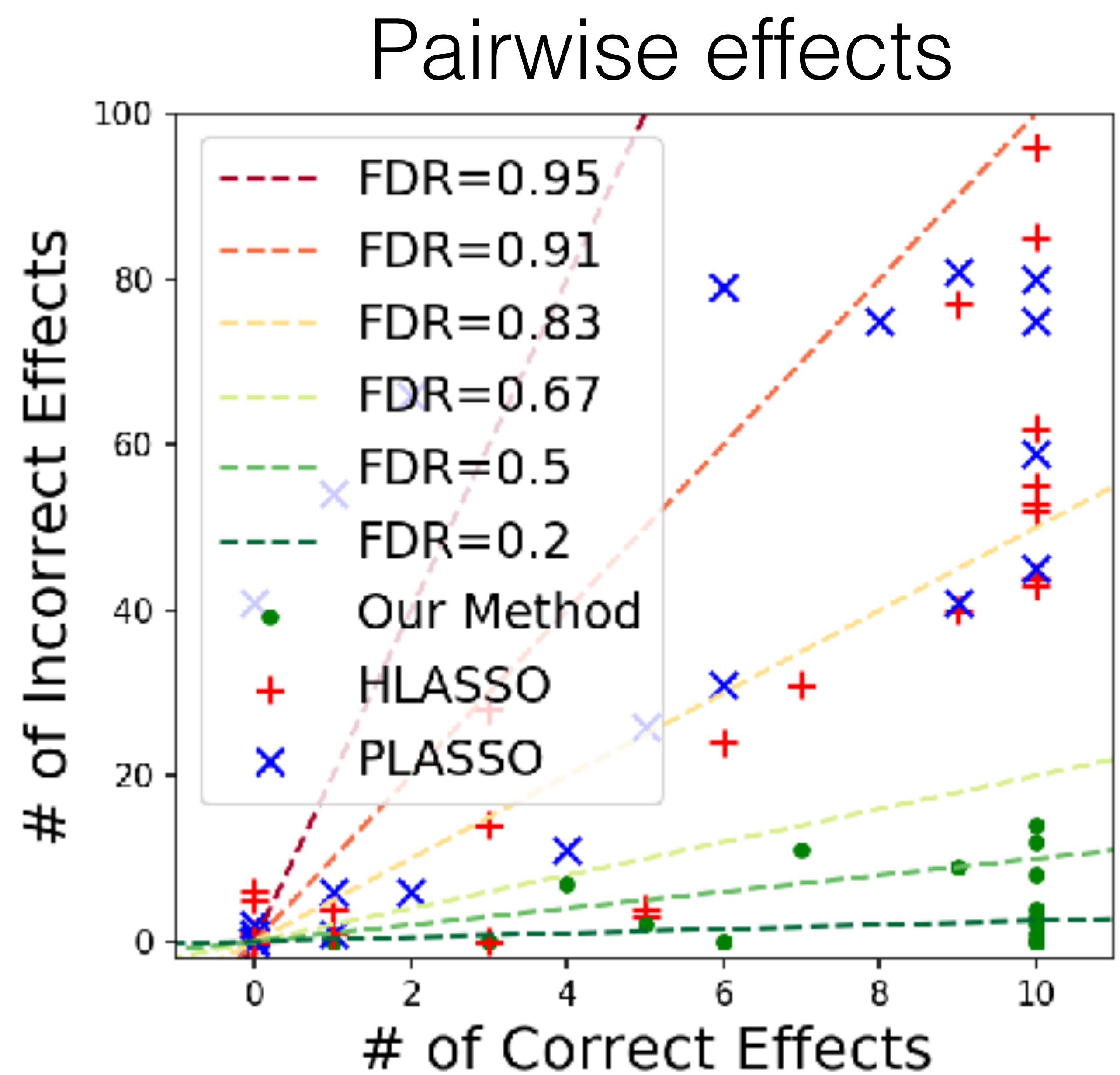
Main effects



Experiments: Simulated, Selection

- 36 different simulated data sets (so know true effects)
  - Up to p = 500 ➔ ≈125,000 total parameters
- False discovery rate (FDR): proportion incorrect





Experiments: Real covariates

Experiments: Real covariates

- Simulated effects: 5 main, 10 interaction

Experiments: Real covariates

- Simulated effects: 5 main, 10 interaction
- Covariates: Residential Building Data Set
  - Highly correlated: 20 of 105 capture 99% of variance

Experiments: Real covariates

- Simulated effects: 5 main, 10 interaction
- Covariates: Residential Building Data Set
  - Highly correlated: 20 of 105 capture 99% of variance
- Key: (# correct effects): (# of incorrect effects)
  - **Higher** green is better: **lower** red is better

Experiments: Real covariates

- Simulated effects: 5 main, 10 interaction
- Covariates: Residential Building Data Set
  - Highly correlated: 20 of 105 capture 99% of variance
- Key: (# correct effects): (# of incorrect effects)
  - **Higher** green is better: **lower** red is better

METHOD	#MAIN	#PAIR
PLASSO	2 : 5	3 : 21

Experiments: Real covariates

- Simulated effects: 5 main, 10 interaction
- Covariates: Residential Building Data Set
  - Highly correlated: 20 of 105 capture 99% of variance
- Key: (# correct effects): (# of incorrect effects)
  - **Higher** green is better: **lower** red is better

Method	#Main	#Pair
PLASSO	2 : 5	3 : 21
HLASSO	3 : 19	3 : 18

Experiments: Real covariates

- Simulated effects: 5 main, 10 interaction
- Covariates: Residential Building Data Set
    - Highly correlated: 20 of 105 capture 99% of variance
- Key: (# correct effects): (# of incorrect effects)
    - **Higher** green is better: **lower** red is better

METHOD	#MAIN	#PAIR
Our method	**3 : 0**	**3 : 0**
PLASSO	**2 : 5**	**3 : 21**
HLASSO	**3 : 19**	**3 : 18**

Experiments: Real data

Experiments: Real data

- Covariates and response: Auto MPG

Experiments: Real data

- Covariates and response: Auto MPG
- N = 398, p = 6 (real-valued), but…

Experiments: Real data

- Covariates and response: Auto MPG
- $N = 398$, $p = 6$ (real-valued), but…
- Augment p with 200 fake (noise) covariates
  - 21,321 total parameters

Experiments: Real data

- Covariates and response: Auto MPG
- N = 398, p = 6 (real-valued), but…
- Augment p with 200 fake (noise) covariates
  - 21,321 total parameters
- Key: (# original effects): (# of fake effects)
  - **No order** to blue: **lower** red is better

Experiments: Real data

- Covariates and response: Auto MPG
- $N = 398$, $p = 6$ (real-valued), but…
- Augment p with 200 fake (noise) covariates
  - 21,321 total parameters
- Key: (# original effects): (# of fake effects)
  - **No order** to blue: **lower** red is better

METHOD	#MAIN	#PAIR
PLASSO	4 : 0	2 : 78

Experiments: Real data

- Covariates and response: Auto MPG
- $N = 398$, $p = 6$ (real-valued), but…
- Augment p with 200 fake (noise) covariates
  - 21,321 total parameters
- Key: (# original effects): (# of fake effects)
  - **No order** to blue: **lower** red is better

Method	#Main	#Pair
PLASSO	**4 : 0**	**2 : 78**
HLASSO	**6 : 46**	**4 : 38**

Experiments: Real data

- Covariates and response: Auto MPG
- N = 398, p = 6 (real-valued), but…
- Augment p with 200 fake (noise) covariates
  - 21,321 total parameters
- Key: (# original effects): (# of fake effects)
  - **No order** to blue: **lower** red is better

Method	#Main	#Pair
Our method	**3 : 0**	**1 : 0**
PLASSO	**4 : 0**	**2 : 78**
HLASSO	**6 : 46**	**4 : 38**

Conclusions

We provide: fast, accurate detection of pairwise interactions

R Agrawal, BL Trippe, JH Huggins, and T Broderick. The Kernel interaction trick: Fast Bayesian discovery of pairwise interactions in high dimensions. *ICML* 2019. ArXiv:1905.06501

Conclusions

We provide: fast, accurate detection of pairwise interactions

R Agrawal, BL Trippe, JH Huggins, and T Broderick. The Kernel interaction trick: Fast Bayesian discovery of pairwise interactions in high dimensions. *ICML* 2019. ArXiv:1905.06501

Thanks to Pyro contributors! Martin Jankowiak, Du Phan, Neeraj Pradhan

- In Pyro: `http://pyro.ai/numpyro/sparse_regression.html`

Conclusions

We provide: fast, accurate detection of pairwise (and higher-order) interactions

R Agrawal, BL Trippe, JH Huggins, and T Broderick. The Kernel interaction trick: Fast Bayesian discovery of pairwise interactions in high dimensions. *ICML* 2019. ArXiv:1905.06501

Thanks to Pyro contributors! Martin Jankowiak, Du Phan, Neeraj Pradhan

- In Pyro: `http://pyro.ai/numpyro/sparse_regression.html`

Conclusions

We provide: fast, accurate detection of pairwise (and higher-order) interactions

Up next:

R Agrawal, BL Trippe, JH Huggins, and T Broderick. The Kernel interaction trick: Fast Bayesian discovery of pairwise interactions in high dimensions. *ICML* 2019. ArXiv:1905.06501

Thanks to Pyro contributors! Martin Jankowiak, Du Phan, Neeraj Pradhan

- In Pyro: `http://pyro.ai/numpyro/sparse_regression.html`

Conclusions

We provide: fast, accurate detection of pairwise (and higher-order) interactions

Up next:

- Response types (binary, count, etc) & nonlinearity

R Agrawal, BL Trippe, JH Huggins, and T Broderick. The Kernel interaction trick: Fast Bayesian discovery of pairwise interactions in high dimensions. *ICML* 2019. ArXiv:1905.06501

Thanks to Pyro contributors! Martin Jankowiak, Du Phan, Neeraj Pradhan

- In Pyro: `http://pyro.ai/numpyro/sparse_regression.html`

Conclusions

We provide: fast, accurate detection of pairwise (and higher-order) interactions

Up next:

- Response types (binary, count, etc) & nonlinearity
- Improve scaling in *N*

R Agrawal, BL Trippe, JH Huggins, and T Broderick. The Kernel interaction trick: Fast Bayesian discovery of pairwise interactions in high dimensions. *ICML* 2019. ArXiv:1905.06501

Thanks to Pyro contributors! Martin Jankowiak, Du Phan, Neeraj Pradhan

- In Pyro: `http://pyro.ai/numpyro/sparse_regression.html`

Conclusions

We provide: fast, accurate detection of pairwise (and higher-order) interactions

Up next:

- Response types (binary, count, etc) & nonlinearity
- Improve scaling in *N*

R Agrawal, BL Trippe, JH Huggins, and T Broderick. The Kernel interaction trick: Fast Bayesian discovery of pairwise interactions in high dimensions. *ICML* 2019. ArXiv:1905.06501

Thanks to Pyro contributors! Martin Jankowiak, Du Phan, Neeraj Pradhan

- In Pyro: `http://pyro.ai/numpyro/sparse_regression.html`

JH Huggins, T Campbell, M Kasprzak, and T Broderick. Scalable Gaussian process inference with finite-data mean and variance guarantees. *AISTATS* 2019.

R Agrawal, T Campbell, JH Huggins, and T Broderick. Data-dependent compression of random features for large-scale kernel approximation. *AISTATS* 2019.

Conclusions

We provide: fast, accurate detection of pairwise (and higher-order) interactions

Up next:

- Response types (binary, count, etc) & nonlinearity
- Improve scaling in N
- Applications!

R Agrawal, BL Trippe, JH Huggins, and T Broderick. The Kernel interaction trick: Fast Bayesian discovery of pairwise interactions in high dimensions. *ICML* 2019. ArXiv:1905.06501

Thanks to Pyro contributors! Martin Jankowiak, Du Phan, Neeraj Pradhan

- In Pyro: `http://pyro.ai/numpyro/sparse_regression.html`

JH Huggins, T Campbell, M Kasprzak, and T Broderick. Scalable Gaussian process inference with finite-data mean and variance guarantees. *AISTATS* 2019.

R Agrawal, T Campbell, JH Huggins, and T Broderick. Data-dependent compression of random features for large-scale kernel approximation. *AISTATS* 2019.

More in the Broderick Group

L Masoero, F Camerlenghi, S Favaro, T Broderick. More for less: Predicting and maximizing variant discovery under a fixed budget via Bayesian nonparametrics. https://arxiv.org/abs/1912.05516

- For fixed budget, there is trade-off in sequencing more genomes and sequencing at greater depth
- We provide new method for prediction of # new variants and optimal allocation of more genomes vs. depth
  - Lowest error when using pilot TCGA dataset to predict the number of new variants to be observed in the follow-up MSK-impact dataset (N=9593) across 197 highly variable, cancerous genes
  - (Only) our prediction can handle when sequencing depth changes between pilot and follow-up study
  - (Only) our method optimizes under fixed budget

T Broderick, R Giordano, R Meager. An Automatic Finite-Sample Robustness Metric: Can Dropping a Little Data Change Conclusions? In preparation.