Background & motivation

Multiresolution dictionary learning

Simulations & applications
For subject $i$ ($i = 1, \ldots, n$), we have a response $y_i \in Y$ & predictors $x_i \in X$. Nonparametric & scalability is key.
For subject \( i \) (\( i = 1, \ldots, n \)), we have a response \( y_i \in \mathcal{Y} \) & predictors \( x_i \in \mathcal{X} \).

Response & predictors can be potentially high-dimensional & complex objects.
For subject $i$ ($i = 1, \ldots, n$), we have a response $y_i \in \mathcal{Y}$ & predictors $x_i \in \mathcal{X}$

Response & predictors can be potentially high-dimensional & complex objects

Our focus: develop flexible & scalable probability models for the conditional distribution $f(y|x)$
General setting

- For subject $i$ ($i = 1, \ldots, n$), we have a response $y_i \in Y$ and predictors $x_i \in X$.
- Response & predictors can be potentially high-dimensional & complex objects.
- **Our focus**: develop flexible & scalable probability models for the conditional distribution $f(y|x)$.
- Nonparametric & scalability is key.
Interaction Example 1: DNA Damage & Repair

Change in shape of DNA damage density with dose + interaction with time for repair
Interaction Example 2: Diabetes Study

2 hour glucose density vs insulin sensitivity ($x_1$) & age ($x_2$)
Interest in predicting creativity from brain imaging data
Interest in predicting creativity from brain imaging data

\( y_i = \) composite creativity index for individual \( i \) \((n = 108)\)
Interest in predicting creativity from brain imaging data

- $y_i =$ composite creativity index for individual $i$ ($n = 108$)
- Diffusion tensor imaging data connectome pipeline used
Interest in predicting creativity from brain imaging data

\[ y_i = \text{composite creativity index for individual } i \ (n = 108) \]

Diffusion tensor imaging data connectome pipeline used

70 vertex undirected weighted brain-graph extracted for each subject
Interest in predicting creativity from brain imaging data

\( y_i = \text{composite creativity index for individual } i \quad (n = 108) \)

Diffusion tensor imaging data connectome pipeline used

70 vertex undirected weighted brain-graph extracted for each subject

\[ p = \binom{70}{2} = 2,415 \text{ edges} \]
Interest in predicting creativity from brain imaging data

\( y_i = \text{composite creativity index for individual } i \ (n = 108) \)

Diffusion tensor imaging data connectome pipeline used

70 vertex undirected weighted brain-graph extracted for each subject

\( p = \binom{70}{2} = 2,415 \) edges

\( x_i = \text{vector of 2,415 connection rates between brain regions} \)
Autism brain imaging data exchange - Yale child study center data ($n = 56$)
Autism brain imaging data exchange - Yale child study center data ($n = 56$)

- Resting state functional MRI processed for analysis of connectomes
Autism brain imaging data exchange - Yale child study center data ($n = 56$)

Resting state functional MRI processed for analysis of connectomes

For each subject, computed measure of normalized power at each voxel (fALFF)
Autism brain imaging data exchange - Yale child study center data \( (n = 56) \)

- Resting state functional MRI processed for analysis of connectomes
- For each subject, computed measure of normalized power at each voxel (fALFF)
- \( fALFF = \text{highly nonlinear transformation of the time series data (} \rho = \text{million)} \)
Autism brain imaging data exchange - Yale child study center data \((n = 56)\)

Resting state functional MRI processed for analysis of connectomes

For each subject, computed measure of normalized power at each voxel (fALFF)

\[ fALFF = \text{highly nonlinear transformation of the time series data} \ (p = \text{million}) \]

\[ y_i = \text{overall head motion (mean frame displacement)} \]
Interested in \( f(y|x) \) for \( y \in \mathcal{Y} \) & \( x = (x_1, \ldots, x_p)' \in \mathcal{X} \)
- Interested in $f(y|x)$ for $y \in \mathcal{Y}$ & $x = (x_1, \ldots, x_p)' \in \mathcal{X}$
- We would like to nonparametrically estimate $f_{Y|X} = \{f(y|x), y \in \mathcal{Y}, x \in \mathcal{X}\}$

Daunting dimensionality problem - statistical & computational bottlenecks

We take a Bayesian approach & choose a prior, $f_{Y|X} \sim \Pi_{Y|X}$, over the space of all possible conditional distributions.
Statistical problem

- Interested in $f(y|x)$ for $y \in \mathcal{Y}$ & $x = (x_1, \ldots, x_p)' \in \mathcal{X}$
- We would like to nonparametrically estimate

$$f_{\mathcal{Y}|\mathcal{X}} = \{f(y|x), y \in \mathcal{Y}, x \in \mathcal{X}\}$$

- Daunting dimensionality problem - statistical & computational bottlenecks
Statistical problem

- Interested in $f(y|x)$ for $y \in \mathcal{Y}$ & $x = (x_1, \ldots, x_p)' \in \mathcal{X}$
- We would like to nonparametrically estimate
  \[ f_{\mathcal{Y}|\mathcal{X}} = \{ f(y|x), y \in \mathcal{Y}, x \in \mathcal{X} \} \]
- Daunting dimensionality problem - statistical & computational bottlenecks
- We take a Bayesian approach & choose a prior,
  \[ f_{\mathcal{Y}|\mathcal{X}} \sim \Pi_{\mathcal{Y}|\mathcal{X}}, \]
  over the space of all possible conditional distributions
Statistical problem

- Interested in $f(y|x)$ for $y \in \mathcal{Y}$ & $x = (x_1, \ldots, x_p)' \in \mathcal{X}$
- We would like to nonparametrically estimate 
  
  $$f_{\mathcal{Y}|\mathcal{X}} = \{f(y|x), y \in \mathcal{Y}, x \in \mathcal{X}\}$$

- Daunting dimensionality problem - statistical & computational bottlenecks
- We take a Bayesian approach & choose a prior, 
  
  $$f_{\mathcal{Y}|\mathcal{X}} \sim \Pi_{\mathcal{Y}|\mathcal{X}}$$

  over the space of all possible conditional distributions
- Current methods have big issues scaling to large $p$
Suppose $x_i = (x_{i1}, \ldots, x_{ip})' \in \mathcal{X} \subset \mathbb{R}^p$ & with the $x_i$'s values concentrated near $\mathcal{M}$
Multiscale manifold learning

- Suppose $x_i = (x_{i1}, \ldots, x_{ip})' \in \mathcal{X} \subset \mathbb{R}^p$ & with the $x_i$s values concentrated near $\mathcal{M}$

- $\mathcal{M}$ is a $d$-dimensional manifold or lower-dimensional subspace
Suppose $x_i = (x_{i1}, \ldots, x_{ip})' \in \mathcal{X} \subset \mathbb{R}^p$ & with the $x_i$s values concentrated near $\mathcal{M}$

$\mathcal{M}$ is a $d$-dimensional manifold or lower-dimensional subspace

At scale $l$ partition $\mathcal{X}$ into $2^l$ mutually exclusive subsets,

$$\mathcal{X} = \bigcup_{h=1}^{2^l} \mathcal{X}_h^{(l)}.$$
Suppose \( x_i = (x_{i1}, \ldots, x_{ip})' \in \mathcal{X} \subset \mathbb{R}^p \) & with the \( x_i \)'s values concentrated near \( \mathcal{M} \)

\( \mathcal{M} \) is a \( d \)-dimensional manifold or lower-dimensional subspace

At scale \( I \) partition \( \mathcal{X} \) into \( 2^I \) mutually exclusive subsets,

\[
\mathcal{X} = \bigcup_{h=1}^{2^I} \mathcal{X}^{(I)}_h.
\]

Very fast multiscale methods are available to estimate the multiscale partition \( \mathcal{P} \)
We estimate the multiscale partition \( \hat{P} \) in a first stage based on only the predictor data.
We estimate the multiscale partition $\hat{P}$ in a first stage based on only the predictor data.

- **metis** (Karypis & Kumar, 1999) - fast multiscale algorithm for graph partitioning - *scalable to huge $p$*
We estimate the multiscale partition $\hat{P}$ in a first stage based on only the predictor data

metis (Karypis & Kumar, 1999) - fast multiscale algorithm for graph partitioning - *scalable to huge* $p$

Essentially a way to simplify/organize the data as a multiscale tree prior to analysis
We estimate the multiscale partition $\hat{P}$ in a first stage based on only the predictor data.

- *metis* (Karypis & Kumar, 1999) - fast multiscale algorithm for graph partitioning - *scalable to huge $p$*

- Essentially a way to simplify/organize the data as a multiscale tree prior to analysis

- We assume the tree encodes the relevant information about the response $y$
We estimate the multiscale partition $\hat{P}$ in a first stage based on only the predictor data.

metis (Karypis & Kumar, 1999) - fast multiscale algorithm for graph partitioning - scalable to huge $p$

Essentially a way to simplify/organize the data as a multiscale tree prior to analysis.

We assume the tree encodes the relevant information about the response $y$.

Trivial to also include coordinates on manifold using GMRA within our approach - we don’t do this for scalability reasons.
Figure 1: (i) Multiscale partition of the data. (ii) Path through the tree for $x_i \in \mathbb{R}^q$. (iii) Conditional density of $y_i$ given $x_i$ defined as a convex combination of densities along the path.
Let \( f_h^{(l)}(y) = N(y; \mu_h^{(l)}, \tau_h^{(l)}) \) denote a dictionary density specific to partition set \( h \) at scale \( l \).
Let $f_h^{(l)}(y) = N(y; \mu_h^{(l)}, \tau_h^{(l)})$ denote a dictionary density specific to partition set $h$ at scale $l$

Estimate $\theta_h^{(l)} = \{\mu_h^{(l)}, \tau_h^{(l)}\}$ via maximum likelihood (ML) using data $\{y_i, i : x_i \in \mathcal{X}_h^{(l)}\}$
Let \( f_h^{(l)}(y) = \mathcal{N}(y; \mu_h^{(l)}, \tau_h^{(l)}) \) denote a dictionary density specific to partition set \( h \) at scale \( l \).

Estimate \( \theta_h^{(l)} = \{\mu_h^{(l)}, \tau_h^{(l)}\} \) via maximum likelihood (ML) using data \( \{y_i, i : x_i \in \mathcal{X}_h^{(l)}\} \).

**OR** obtain posterior for \( \theta_h^{(l)} \) updating normal-gamma prior with data \( \{y_i, i : x_i \in \mathcal{X}_h^{(l)}\} \).
Let $f_h^{(l)}(y) = N(y; \mu_h^{(l)}, \tau_h^{(l)})$ denote a dictionary density specific to partition set $h$ at scale $l$.

Estimate $\theta_h^{(l)} = \{\mu_h^{(l)}, \tau_h^{(l)}\}$ via maximum likelihood (ML) using data $\{y_i, i : x_i \in \mathcal{X}_h^{(l)}\}$.

OR obtain posterior for $\theta_h^{(l)}$ updating normal-gamma prior with data $\{y_i, i : x_i \in \mathcal{X}_h^{(l)}\}$.

A CART-type approach would let $f(y|x) = f_h^{(L)}(y)$ with $L$ the finest (leaf) scale for $x \in \mathcal{X}_h^{(L)}$. 

Problem: few individuals allocated to any particular leaf partition - low bias but big variance
Problem: few individuals allocated to any particular leaf partition - low bias but big variance

We instead borrow information across different resolutions of the tree using a nonparametric Bayes approach.
Making better use of the dictionary - MSE considerations

- **Problem:** few individuals allocated to any particular leaf partition - low bias but big variance
- We instead borrow information across different resolutions of the tree using a nonparametric Bayes approach
- Instead of \( f(y|x) = f_h^{(L)}(y) \) we use

\[
 f(y|x) = \sum_{l=1}^{L} \pi_{h_l(x)}^{(l)} f_h^{(l)}(y),
\]

where \( h(x) = \{h_1(x), \ldots, h_L(x)\} = \text{path through } \hat{P} \text{ specific to } x \)
Figure 1: (i) Multiscale partition of the data. (ii) Path through the tree for $x_i \in \mathbb{R}^q$. (iii) Conditional density of $y_i$ given $x_i$ defined as a convex combination of densities along the path.
At the coarse scales of resolution, more subjects will be allocated to each partition set, leading to lower variance.
At the coarse scales of resolution, more subjects will be allocated to each partition set, leading to lower variance.

Reduction in variance is at the expense of bias **IF** the finer scales are needed to characterize the true $f_0(y|x)$. 

It is not clear a priori how to distribute probability over resolutions, but it should depend on path through the tree.

We learn a posterior on the appropriate multiresolution probability weights.
At the coarse scales of resolution, more subjects will be allocated to each partition set, leading to lower variance.

Reduction in variance is at the expense of bias if the finer scales are needed to characterize the true $f_0(y|x)$.

Not clear a priori how to distribute probability over resolutions, but it should depend on path through the tree.
At the coarse scales of resolution, more subjects will be allocated to each partition set, leading to lower variance.

Reduction in variance is at the expense of bias IF the finer scales are needed to characterize the true $f_0(y|x)$.

Not clear a priori how to distribute probability over resolutions, but it should depend on path through the tree.

We learn a posterior on the appropriate multiresolution probability weights.
Our multiresolution stick-breaking process lets

\[ \pi_h(x) = V_h(x) \prod_{m=1}^{l-1} (1 - V_{h_m}(x)), \quad V_h \sim \text{Be}(1, \alpha). \]
Our multiresolution stick-breaking process lets

\[ \pi_{h_l(x)} = V_{h_l(x)} \prod_{m=1}^{l-1} (1 - V_{h_m(x)}^{(m)}), \quad V_{h_l} \sim \text{Be}(1, \alpha). \]

The unit probability stick is used up as we go from a coarse to finer resolution.
Our multiresolution stick-breaking process lets

\[
\pi_h^{(l)}(x) = V_h^{(l)} \prod_{m=1}^{l-1} (1 - V_h^{(m)}(x)), \quad V_h^{(l)} \sim \text{Be}(1, \alpha).
\]

The unit probability stick is used up as we go from a coarse to finer resolution.

Data can inform strongly about how fast we use up the stick in different regions of the tree.
Our multiresolution stick-breaking process lets

\[
\pi^{(l)}_{h_l(x)} = V^{(l)}_{h_l(x)} \prod_{m=1}^{l-1} (1 - V^{(m)}_{h_m(x)}), \quad V^{(l)}_h \sim \text{Be}(1, \alpha).
\]

The unit probability stick is used up as we go from a coarse to finer resolution.

Data can inform strongly about how fast we use up the stick in different regions of the tree.

Induces effective dimensionality reduction - in extreme case, posterior for \( \alpha \) can concentrate near zero so \( f(y|x) \approx f^{(1)}(y) \).
Our multiresolution stick-breaking process lets

\[ \pi_{h_l(x)}^{(l)} = V_{h_l(x)}^{(l)} \prod_{m=1}^{l-1} (1 - V_{h_m(x)}^{(m)}) , \quad V_{h}^{(l)} \sim \text{Be}(1, \alpha). \]

The unit probability stick is used up as we go from a coarse to finer resolution.

Data can inform strongly about how fast we use up the stick in different regions of the tree.

Induces effective dimensionality reduction - in extreme case, posterior for \( \alpha \) can concentrate near zero so \( f(y|x) \approx f^{(1)}(y) \)

Favors similarity in \( f(y|x) \) and \( f(y|x') \) for close \( x \) & \( x' \) - even if not in same leaf node.
Motivation: practical performance weighting heavily computational scalability
Motivation: practical performance weighting heavily computational scalability

I’ve got tons of applications with huge $p$ and want something that works great practically
Computation

- **Motivation**: practical performance weighting heavily computational scalability
- I’ve got tons of applications with huge $p$ and want something that works great practically
- To implement, we pre-compute $\hat{P}$ using fast algorithms
Computation

- **Motivation**: practical performance weighting heavily computational scalability
- I’ve got tons of applications with huge $p$ and want something that works great practically
- To implement, we pre-compute $\hat{P}$ using fast algorithms
- We then do ML or conjugate Bayes updating to get the multiresolution dictionary of densities
Computation

- **Motivation**: practical performance weighting heavily computational scalability
- I’ve got tons of applications with huge $p$ and want something that works great practically
- To implement, we pre-compute $\hat{P}$ using fast algorithms
- We then do ML or conjugate Bayes updating to get the multiresolution dictionary of densities
- Fixing $\hat{P}$ & the dictionary, we run slice sampling for posterior computation of the stick-breaking component
**Motivation**: practical performance weighting heavily computational scalability

I’ve got tons of applications with huge $p$ and want something that works great practically

To implement, we pre-compute $\hat{P}$ using fast algorithms

We then do ML or conjugate Bayes updating to get the multiresolution dictionary of densities

Fixing $\hat{P}$ & the dictionary, we run slice sampling for posterior computation of the stick-breaking component

Very fast & can implement the analysis in our motivating data sets in a minute or two, while competitors (e.g. random forests) break down
Early stopping of Gibbs sampler - Chauveau & Diebolt (98)

- Predictors in $r$-dimensional subspace (linear or non-linear)
- Dimension of predictors is HUGE - up to $p = 1,000,000$ with $n \in \{50, 100, 200\}$.
- Compared to CART, Lasso, random forests (RF) - RF too slow
- In every case (among many) we are similar to substantially better than Lasso & CART in MSE
- Substantially faster & produce non-Gaussian conditional density estimates with uncertainty intervals
Simulation Study

- Early stopping of Gibbs sampler - Chauveau & Diebolt (98)

- Predictors in $r$-dimensional subspace (linear or non-linear) with $r$ small
Simulation Study

- Early stopping of Gibbs sampler - Chauveau & Diebolt (98)
- Predictors in $r$-dimensional subspace (linear or non-linear) with $r$ small
- Dimension of predictors is HUGE - up to $p = 1,000,000$ with $n \in \{50, 100, 200\}$. Compared to CART, Lasso, random forests (RF) - RF too slow
- In every case (among many) we are similar to substantially better than Lasso & CART in MSE
- Substantially faster & produce non-Gaussian conditional density estimates with uncertainty intervals
Early stopping of Gibbs sampler - Chauveau & Diebolt (98)

Predictors in $r$-dimensional subspace (linear or non-linear) with $r$ small

Dimension of predictors is HUGE - up to $p = 1,00,000+$ with $n \in \{50, 100, 200\}$.

Compared to CART, Lasso, random forests (RF) - RF too slow
Early stopping of Gibbs sampler - Chauveau & Diebolt (98)

Predictors in $r$-dimensional subspace (linear or non-linear) with $r$ small

Dimension of predictors is HUGE - up to $p = 1,00,000+$ with $n \in \{50, 100, 200\}$.

Compared to CART, Lasso, random forests (RF) - RF too slow

In every case (among many) we are similar to substantially better than Lasso & CART in MSE
Early stopping of Gibbs sampler - Chauveau & Diebolt (98)
Predictors in $r$-dimensional subspace (linear or non-linear) with $r$ small
Dimension of predictors is **HUGE** - up to $p = 1,000\text{+}$ with $n \in \{50, 100, 200\}$.
Compared to CART, Lasso, random forests (RF) - RF too slow
In every case (among many) we are similar to substantially better than Lasso & CART in MSE
Substantially faster & produce non-Gaussian conditional density estimates with uncertainty intervals
Simulation 1 - $p = 1000$, $n = (a)100, (b)150, (c)200$ at $x$ points I & II

truth - 1d non-linear subspace with response mixing normals depending on subspace coords
Simulation 2 - truth is linear subspace

(I) ratio of mses (MSB numerator); (II) CPU times (sec) - solid CART, Lash (dash), MSB (dash-dot)
Simulation 3 - nonlinear subspace, $p = 300,000$, $n$ varies

Boxplots of (I) $t_{mse}$ as $p$ increases and (II) $t_{cpu}$ - data drawn from mixture of factor analyzers
Boxplots of $t_{mse}$ as $p$ increases
Simulation 4 - swiss roll simulation
Measurements of creativity for 108 subjects
Neuroscience Application 1

- Measurements of creativity for 108 subjects
- For each subject, extract a brain graph of 70 cortical regions, whose centers are vertices

\[
p = 2,415
\]

MSB lowest MSE (0.56) & fastest computing time (100 second - leave one out CV)

RF MSE = 0.57 but took 7,817 sec, CART failed to estimate any signal & Lasso had MSE=0.63 and time = 50 sec.
Neuroscience Application 1

- Measurements of creativity for 108 subjects
- For each subject, extract a brain graph of 70 cortical regions, whose centers are vertices
- Count # connections between different regions
Measurements of creativity for 108 subjects

For each subject, extract a brain graph of 70 cortical regions, whose centers are vertices

Count # connections between different regions

predictors = log # connections between each pair of vertices ($p = 2,415$)
Neuroscience Application 1

- Measurements of creativity for 108 subjects
- For each subject, extract a brain graph of 70 cortical regions, whose centers are vertices
- Count # connections between different regions
- Predictors = log # connections between each pair of vertices ($p = 2,415$)
- MSB lowest MSE (0.56) & fastest computing time (100 second - leave one out CV)

RF MSE = 0.57 but took 7,817 sec, CART failed to estimate any signal & Lasso had MSE=0.63 and time = 50 sec.
Measurements of creativity for 108 subjects
For each subject, extract a brain graph of 70 cortical regions, whose centers are vertices
Count # connections between different regions
predictors $= \log$ # connections between each pair of vertices ($p = 2,415$)
MSB lowest MSE (0.56) & fastest computing time (100 second - leave one out CV)
RF MSE $= 0.57$ but took 7,817 sec, CART failed to estimate any signal & Lasso had MSE $= 0.63$ and time $= 50$ sec.
Much larger $p = 1,000,000$ with $n = 56$
Much larger $p = 1,000,000$ with $n = 56$

Predict head motion from connectome features measured in fMRI
- Much larger $p = 1,000,000$ with $n = 56$
- Predict head motion from connectome features measured in fMRI
- Only Lasso & MSB could be implemented give huge $p$
- Much larger $p = 1,000,000$ with $n = 56$
- Predict head motion from connectome features measured in fMRI
- Only Lasso & MSB could be implemented give huge $p$
- Lasso failed to detect any signal - MSE = 1.02 (null model has 1.00) & computing time is 5,836 sec
Much larger $p = 1,000,000$ with $n = 56$

Predict head motion from connectome features measured in fMRI

Only Lasso & MSB could be implemented give huge $p$

Lasso failed to detect any signal - MSE = 1.02 (null model has 1.00) & computing time is 5,836 sec

MSB has MSE = 0.76 & computing time = 690 seconds
Much larger $p = 1,000,000$ with $n = 56$

Predict head motion from connectome features measured in fMRI

Only Lasso & MSB could be implemented give huge $p$

Lasso failed to detect any signal - MSE = 1.02 (null model has 1.00) & computing time is 5,836 sec

MSB has MSE = 0.76 & computing time = 690 seconds

Unlike Lasso we aren’t choosing any tuning parameters by CV
Mse & times for n-fold CV for neuroscience applications.

<table>
<thead>
<tr>
<th>DATA</th>
<th>n</th>
<th>p</th>
<th>MODEL</th>
<th>MSE</th>
<th>(t_T)</th>
<th>(t_M)</th>
<th>(t_V)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1)</td>
<td>108</td>
<td>2,415</td>
<td>MSB</td>
<td>0.56</td>
<td>100</td>
<td>1.1</td>
<td>0.02</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CART</td>
<td>1.10</td>
<td>87</td>
<td>0.9</td>
<td>0.01</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>LASSO</td>
<td>0.63</td>
<td>50</td>
<td>0.40</td>
<td>0.10</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>RF</td>
<td>0.57</td>
<td>7,817</td>
<td>78.2</td>
<td>0.59</td>
</tr>
<tr>
<td>(2)</td>
<td>56</td>
<td>1e+05</td>
<td>MSB</td>
<td>0.76</td>
<td>690</td>
<td>20.98</td>
<td>2.31</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>LASSO</td>
<td>1.02</td>
<td>5,836</td>
<td>96.18</td>
<td>9.66</td>
</tr>
</tbody>
</table>

\(t_T\) = time for all subject predictions, \(t_M\) (\(t_V\)) = mean (st dev) of time needed across subjects
Focus on estimating conditional distributions of response variables given high-dimensional predictors
Focus on estimating conditional distributions of response variables given high-dimensional predictors

Proposed (partly) Bayes multiresolution approach - great scalability & practical performance in applications we’ve considered

Interesting to obtain theoretical guarantees for such hybrid frequentist-Bayes methods

Bayes methods need to become scalable to be relevant in modern applications
Discussion

- Focus on estimating conditional distributions of response variables given high-dimensional predictors.
- Proposed (partly) Bayes multiresolution approach - great scalability & practical performance in applications we’ve considered.
- Focused on univariate continuous response but trivial to modify to classification & multivariate conditional densities.
Focus on estimating conditional distributions of response variables given high-dimensional predictors

Proposed (partly) Bayes multiresolution approach - great scalability & practical performance in applications we’ve considered

Focused on univariate continuous response but trivial to modify to classification & multivariate conditional densities

Interesting to obtain theoretical guarantees for such hybrid frequentist-Bayes methods
Focus on estimating conditional distributions of response variables given high-dimensional predictors

Proposed (partly) Bayes multiresolution approach - great scalability & practical performance in applications we’ve considered

Focused on univariate continuous response but trivial to modify to classification & multivariate conditional densities

Interesting to obtain theoretical guarantees for such hybrid frequentist-Bayes methods

Bayes methods need to become scalable to be relevant in modern applications