Using Profile Regression Mixture Models and Dirichlet Processes to explore the combined effect of risk factors; the R package PReMiuM

Silvia Liverani and Michail Paphthomas

British-German meeting on classification - UCL
November 2013
Outline

- Motivation
- Method
- The R package PReMiuM
- Examples
People

- David Hastie
- John Molitor
- Sylvia Richardson
Motivation

Multicollinearity

- Goal of epidemiological studies is to investigate the joint effect of different covariates / risk factors on a phenotype...
- ... but highly correlated risk factors create collinearity problems!
Motivation

Multicollinearity

- Goal of epidemiological studies is to investigate the joint effect of different covariates / risk factors on a phenotype...
- ... but highly correlated risk factors create collinearity problems!

Example

Researchers are interested in determining if a relationship exists between **blood pressure** \((y = \text{BP}, \text{in mm Hg})\) and

- **weight** \((x_1 = \text{Weight, in kg})\)
- **body surface area** \((x_2 = \text{BSA, in sq m})\)
- duration of hypertension \((x_3 = \text{Dur, in years})\)
- basal pulse \((x_4 = \text{Pulse, in beats per minute})\)
- stress index \((x_5 = \text{Stress})\)
Weight = $x_1$, BSA = $x_2$
Highly correlated risk factors create collinearity problems, causing instability in model estimation.

<table>
<thead>
<tr>
<th>Model</th>
<th>$\hat{\beta}_1$</th>
<th>SE $\hat{\beta}_1$</th>
<th>$\hat{\beta}_2$</th>
<th>SE $\hat{\beta}_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$y \sim x_1$</td>
<td>2.64</td>
<td>0.30</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>$y \sim x_2$</td>
<td>–</td>
<td>–</td>
<td>3.34</td>
<td>1.33</td>
</tr>
<tr>
<td>$y \sim x_1 + x_2$</td>
<td>6.58</td>
<td>0.53</td>
<td>-20.44</td>
<td>2.28</td>
</tr>
</tbody>
</table>
Highly correlated risk factors create collinearity problems, causing instability in model estimation.

<table>
<thead>
<tr>
<th>Model</th>
<th>$\hat{\beta}_1$</th>
<th>SE $\hat{\beta}_1$</th>
<th>$\hat{\beta}_2$</th>
<th>SE $\hat{\beta}_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$y \sim x_1$</td>
<td>2.64</td>
<td>0.30</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>$y \sim x_2$</td>
<td>–</td>
<td>–</td>
<td>3.34</td>
<td>1.33</td>
</tr>
<tr>
<td>$y \sim x_1 + x_2$</td>
<td>6.58</td>
<td>0.53</td>
<td>-20.44</td>
<td>2.28</td>
</tr>
</tbody>
</table>

**Effect 1**: the estimated regression coefficient of any one variable depends on which other predictor variables are included in the model.

**Effect 2**: the precision of the estimated regression coefficients decreases as more predictor variables are added to the model.
When it is of interest to detect interactions between covariates, standard regression modelling may become problematic.

- In a classical setting, fitting a linear model with many parameters sometimes requires an impractically large vector of observations to produce valid inferences (Burton et al., IJE, 2009). Also, identifiability and collinearity problems are often present.

- In Bayesian model comparison, the space of models becomes vast, and model search algorithms like the Reversible Jump approach (Green, Bka 1995) require an impractically large number of iterations before they converge (Dobra and Massam, St Meth 2010).
Issues caused by

- correlated risk factors
- interacting risk factors
Issues caused by

- correlated risk factors
- interacting risk factors

Profile regression

- partitions the subjects into groups according to covariate profile
- investigation of the joint effects of multiple risk factors
- jointly models the covariate patterns and health outcomes
- flexible but tractable Bayesian model
$x_1$ and $x_2$

only $x_1$
Notation

For individual \( i \)

- \( y_i \)  
  - outcome of interest
- \( x_i = (x_{i1}, \ldots, x_{iP}) \)  
  - covariate profile
- \( w_i \)  
  - fixed effects
- \( z_i = c \)  
  - the allocation variable indicates the cluster to which individual \( i \) belongs
Statistical Framework

- Mixture model for the covariates

\[
f(x_i|\phi, \psi) = \sum_c \psi_c f(x_i|z_i = c, \phi_c)
\]
Statistical Framework

- Mixture model for the covariates

\[ f(x_i | \phi, \psi) = \sum_c \psi_c f(x_i | z_i = c, \phi_c) \]

- For example for discrete covariates

\[ f(x_i | z_i = c, \phi_c) = \prod_{j=1}^{J} \phi_{z_i,j,x_i,j} \]
Statistical Framework

- Mixture model for the covariates

\[
f(\mathbf{x}_i|\phi, \psi) = \sum_c \psi_c f(\mathbf{x}_i|z_i = c, \phi_c)
\]

- For example for discrete covariates

\[
f(\mathbf{x}_i|z_i = c, \phi_c) = \prod_{j=1}^{J} \phi_{z_i,j,\mathbf{x}_i,j}
\]
Mixture model for the covariates

\[ f(x_i | \phi, \psi) = \sum_{c} \psi_c f(x_i | z_i = c, \phi_c) \]
Statistical Framework

- Mixture model for the covariates

\[ f(x_i | \phi, \psi) = \sum_c \psi_c f(x_i | z_i = c, \phi_c) \]

- Prior model for the mixture weights \( \psi_c \)
  - stick-breaking priors (constructive definition of the Dirichlet Process)

\[ \mathbb{P}(Z_i = c | \psi) = \psi_c \]
Statistical Framework

- Mixture model for the covariates

\[ f(x_i | \phi, \psi) = \sum_c \psi_c f(x_i | z_i = c, \phi_c) \]

- Prior model for the mixture weights \( \psi_c \)
  - stick-breaking priors (constructive definition of the Dirichlet Process)

\[ P(Z_i = c | \psi) = \psi_c \quad \psi_1 = V_1 \]

\[ \psi_c = V_c \prod_{l < c} (1 - V_l) \quad V_c \sim \text{Beta}(1, \alpha) \]
Statistical Framework

- Mixture model for the covariates
  \[
  f(x_i | \phi, \psi) = \sum_c \psi_c f(x_i | z_i = c, \phi_c)
  \]

- Prior model for the mixture weights \( \psi_c \)
  - stick-breaking priors (constructive definition of the Dirichlet Process)
    \[
    \mathbb{P}(Z_i = c | \psi) = \psi_c \quad \psi_1 = V_1
    \]
    \[
    \psi_c = V_c \prod_{l < c} (1 - V_l) \quad V_c \sim \text{Beta}(1, \alpha)
    \]
  - larger concentration parameter \( \alpha \) the more evenly distributed is the resulting distribution.
  - smaller concentration parameter \( \alpha \) the more sparsely distributed is the resulting distribution, with all but a few parameters having a probability near zero
**Statistical Framework**

- Joint covariate and response model

\[
f(x_i, y_i | \phi, \theta, \psi, \beta) = \sum_c \psi_c f(x_i | z_i = c, \phi_c) f(y_i | z_i = c, \theta_c, \beta, w_i)
\]
Statistical Framework

- Joint covariate and response model

\[
f(x_i, y_i|\phi, \theta, \psi, \beta) = \sum_c \psi_c f(x_i|z_i = c, \phi_c) f(y_i|z_i = c, \theta_c, \beta, w_i)
\]

- Mixture model jointly for covariate and response

- For example, for Bernoulli outcome

\[
\text{logit}\{p(y_i = 1|\theta_c, \beta, w_i)\} = \theta_c + \beta^T w_i
\]
Statistical Framework

- Joint covariate and response model

\[ f(x_i, y_i | \phi, \theta, \psi, \beta) = \sum_c \psi_c f(x_i | z_i = c, \phi_c) f(y_i | z_i = c, \theta_c, \beta, w_i) \]

- Mixture model jointly for covariate and response

- For example, for Bernoulli outcome

\[ \logit\{ p(y_i = 1 | \theta_c, \beta, w_i) \} = \theta_c + \beta^T w_i \]

- The association of the profiles with the response are characterised by the risk effect parameters \( \theta_c \)
Statistical Framework

- Joint covariate and response model

\[ f(x_i, y_i | \phi, \theta, \psi, \beta) = \sum_c \psi_c f(x_i | z_i = c, \phi_c) f(y_i | z_i = c, \theta_c, \beta, w_i) \]

- Mixture model jointly for covariate and response

- For example, for Bernoulli outcome

\[ \logit\{p(y_i = 1 | \theta_c, \beta, w_i)\} = \theta_c + \beta^T w_i \]

- The association of the profiles with the response are characterised by the risk effect parameters \( \theta_c \)

- Note, the above framework, adopted in Molitor et al. (Biostatistics, 2010) is similar to the one in Bigelow and Dunson (JASA, 2009).
We have implemented profile regression in C++ (now wrapped in an R package) for
- binary, binomial, categorical, Normal and Poisson outcome
- Normal and discrete covariates

It can do
- dependent or independent slice sampling (Kalli et al., 2011)
- or truncated Dirichlet process model (Ishwaran and James, 2001)

as well as
- handles missing data
  - we check which cluster the subject is allocated to and then sample
- includes some label switching moves
Characterising the “best partition”

- Construct a score matrix $S = S_{ij}$ that records the probability that individuals $i$ and $j$ are assigned to the same cluster.
- Task is to find a partition $z^{\text{best}}$ that best represents $S$
  - susceptible to Monte Carlo error
  - Dahl (2006) suggests using a least square distance criterion
  - Alternatively, we process the similarity matrix through the Partitioning Around Medoids algorithm (Kaufman & Rousseu, 1990) to derive $z^{\text{best}}$. 
Once $z^{best}$ is defined, it is important to have a measure of uncertainty/stability.

- This can be done by model averaging quantities related to $z^{best}$ (effects and cluster related parameters) through post-processing.
- If $z^{best}$ is a consistent clustering, then subgroup parameters will vary little from iteration to iteration, leading to narrow posterior credible intervals.

**Predicting**

- At each iteration of the sampler, calculate the probability that $i'$ belongs to each of the clusters:

$$p(z_{i'} = c|x_{i'}, \psi_c, \phi) \propto p(z_{i'} = c, \psi_c)p(x_{i'} | z_{i'} = c, \phi)$$

and sample $\theta_c$ according to these probabilities.
R package PReMiuM

Program to implement Dirichlet Process Bayesian Clustering

- Allows user to exclude the response from the model
- Allows user to run with a fixed alpha or update alpha (default)
- Allows users to run predictive scenarios (basic or Rao-Blackwellised predictions)
- Handling of missing data
- Adaptive MCMC where appropriate
- Can generate simulation data
- Performs post processing
- Functions for flexible plotting
R package PReMiuM

- Can be downloaded from CRAN
  http://cran.r-project.org/web/packages/PReMiuM/
- or

  `install.packages("PReMiuM")`
Example: Simulated data

The profiles are given by

- \( y \) : outcome, Bernoulli
- \( x \) : 5 covariates, all discrete with 3 levels
- \( w \) : 2 fixed effects, continuous or discrete
Example: Simulated data

The profiles are given by

- \( y \) : outcome, Bernoulli
- \( x \) : 5 covariates, all discrete with 3 levels
- \( w \) : 2 fixed effects, continuous or discrete

```r
> profRegr(yModel="Bernoulli",
          xModel="Discrete",
          nSweeps=1000,
          nBurn=1000,
          data="dataMatrix.txt",
          output="output/output",
          nCovariates=5,
          covNames=c("Var1","Var2","Var3","Var4","Var5"),
          nFixedEffects=2,
          fixedEffectsNames=c("FE1","FE2"),
          xLevels=c(3,3,3,3,3))
```
R package DiPBaC: Output

The programme creates a number of output files:

- `output_z.txt` → allocations $z$

- `output_nClusters.txt` → cluster sizes

- `output_theta.txt` → cluster-specific parameter for outcome: $\theta$

- `output_phi.txt` → cluster-specific parameter for covariates: $\phi$

- `output_alpha.txt` → parameter of the Dirichlet process $\alpha$

- `output_beta.txt` → fixed effects parameter $\beta$

- `output_entropy.txt`

- `output_nMembers.txt`

- `output_logPost.txt`

- `output_psi.txt`

- `output_betaProp.txt`

- `output_log.txt`

- `output_thetaProp.txt`

- `output_alphaProp.txt`

- `output_input.txt`
The programme creates a number of output files:

- **output_z.txt → allocations \( z \)**
- **output_nClusters.txt → cluster sizes**
R package DiPBaC: Output

The programme creates a number of output files:

- `output_z.txt` → allocations $z$
- `output_nClusters.txt` → cluster sizes
- `output_theta.txt` → cluster-specific parameter for outcome: $\theta$

- `output_phi.txt` → cluster-specific parameter for covariates: $\phi$
- `output_alpha.txt` → parameter of the Dirichlet process $\alpha$
- `output_beta.txt` → fixed effects parameter $\beta$
- `output_entropy.txt`
- `output_nMembers.txt`
- `output_logPost.txt`
- `output_psi.txt`
- `output_betaProp.txt`
- `output_log.txt`
- `output_thetaProp.txt`
- `output_alphaProp.txt`
- `output_input.txt`
R package DiPBaC: Output

The programme creates a number of output files:

- `output_z.txt` → allocations \( z \)
- `output_nClusters.txt` → cluster sizes
- `output_theta.txt` → cluster-specific parameter for outcome: \( \theta \)
- `output_phi.txt` → cluster-specific parameter for covariates: \( \phi \)
R package DiPBaC: Output

The programme creates a number of output files:

- `output_z.txt` → allocations $z$
- `output_nClusters.txt` → cluster sizes
- `output_theta.txt` → cluster-specific parameter for outcome: $\theta$
- `output_phi.txt` → cluster-specific parameter for covariates: $\phi$
- `output_alpha.txt` → parameter of the Dirichlet process $\alpha$
R package DiPBaC: Output

The programme creates a number of output files:

- `output_z.txt` → allocations $z$
- `output_nClusters.txt` → cluster sizes
- `output_theta.txt` → cluster-specific parameter for outcome: $\theta$
- `output_phi.txt` → cluster-specific parameter for covariates: $\phi$
- `output_alpha.txt` → parameter of the Dirichlet process $\alpha$
- `output_beta.txt` → fixed effects parameter $\beta$
The programme creates a number of output files:

- **output_z.txt** → allocations \( z \)
- **output_nClusters.txt** → cluster sizes
- **output_theta.txt** → cluster-specific parameter for outcome: \( \theta \)
- **output_phi.txt** → cluster-specific parameter for covariates: \( \phi \)
- **output_alpha.txt** → parameter of the Dirichlet process \( \alpha \)
- **output_beta.txt** → fixed effects parameter \( \beta \)
- **output_entropy.txt**
- **output_nMembers.txt**
- **output_logPost.txt**
- **output_psi.txt**
- **output_betaProp.txt**
- **output_log.txt**
- **output_thetaProp.txt**
- **output_alphaProp.txt**
- **output_input.txt**
R package PReMiuM: Postprocessing

- Returns file with output info
  ```
  > runInfoObj <- readRunInfo(fileStem = "output", directoryPath = "output")
  ```

- Computes the score matrix that records the probability that each pair of individuals are in the same clusters
  ```
  > dissimObj <- calcDissimilarityMatrix(runInfoObj)
  ```

- Computes the optimal partition
  ```
  > clusObj <- calcOptimalClustering(dissimObj)
  ```
R package PReMiuM: Postprocessing

- Computes the average risk and profile for each cluster
  ```r
  riskProfileObj <- calcAvgRiskAndProfile(clusObj)
  ```
- Plots
  ```r
  clusterOrderObj <- plotRiskProfile(riskProfileObj, "output/summary.png")
  ```
Predictions

```r
> profRegr(yModel="Bernoulli",
         xModel="Discrete",
         nSweeps=1000,
         nBurn=1000,
         data="dataMatrix.txt",
         output="output/output",
         nCovariates=5,
         covNames=c("Var1","Var2","Var3","Var4","Var5"),
         nFixedEffects=2,
         fixedEffectsNames=c("FE1","FE2"),
         xLevels=c(3,3,3,3,3),
         predict="predict.scenarios.txt")
```
The file `predict.scenarios.txt` contains the predictive scenarios for the 5 covariates and the 2 fixed effects.

0 1 0 0 0 -0.348608422340999 0.786982503814148
0 0 1 0 0 3.832601433577385 -0.67410633721989

Postprocessing follows as before

Predictions are postprocessed separately as follows

```r
> predictions <- calcPredictions(riskProfileObj,
                                 doRaoBlackwell=F,
                                 fullSweepPredictions=T,
                                 fullSweepLogOR=T)
```
Variable selection

- In many applications, the number of predictors is large and the full profiles become difficult to interpret
  - Aim is to identify the predictors that contribute more than others to the formation of sub-populations

- Use statistical procedures embedding variable selection within profile regression (Papathomas et al., 2012), inspired from Tadesse et al. (2005) and Chung and Dunson (2009)
Variable selection

- Context of application: genetic epidemiology where there is interest in investigating “gene-gene interactions”
  - As the number of markers genotyped is huge, blind search for interactions followed by adjustment for multiple testing has very low power → more focussed strategies are recommended

- In MP et al. (2012) we investigate how profile regression combined with variable selection can highlight combinations of SNPs associated with higher disease risk in a genetic association study of lung cancer (Hung et al 2008).
Using 0-1 variable selection switches

- Consider cluster specific binary switches, $\gamma_c, j$, so that
  - $\gamma_c, j = 1$ when predictor $x_j$ is important for allocating subjects to cluster $c$, and in this case the associated probabilities are $\phi_c, j(k), 1 \leq k \leq m_j$
  - When $\gamma_c, j = 0$ the associated probability is $\phi_0, j(k)$, a common probability for all individuals for whom $x_{ij} = k$.

So the discrete covariate model is modified accordingly to

$$f(x_i | \phi_c, \gamma_c) = \prod_{j: \gamma_c, j=1} \phi_{z_i, j(x_{ij})} \prod_{j: \gamma_c, j=0} \phi_{0, j(x_{ij})}$$

- Prior for switches: given $\rho_j$, $\gamma_c, j \sim \text{Bernoulli}(1, \rho_j)$.

- We consider a sparsity inducing prior for $\rho$ with an atom at zero:

  $$\rho_j \sim 1\{w_j=0\} \delta_0(\rho_j) + 1\{w_j=1\} \text{Beta}(\alpha_\rho, \beta_\rho)$$

  where $w_j \sim \text{Bernoulli}(0.5)$.

Similar to Chung and Dunson (JASA, 2009), but in their set up, covariate observations contribute to the likelihood through a regression model. In our case, covariate observations contribute directly to the likelihood, and we introduce $\phi_{0,j}(x_{ij})$. 

The R package PReMiuM

Using latent selection probabilities

- The 0-1 variable selection switches model exhibits sometimes poor mixing, requiring long runs.
- We introduce an new parametrisation involving continuous selection probabilities $\zeta_j$, taking values in $[0, 1]$.
- Each $\phi_{c,j}$ is replaced by a hybrid parameter

\[
\phi^*_c, j(x) = \zeta_j \times \phi_{c,j}(x) + (1 - \zeta_j) \times p_j(x)
\]

where $p_j(k)$ is the observed proportion of individuals for whom $x_{ij} = k$.

- Parameters $\zeta_j$ can be viewed as proxies for the probability that predictor $j$ supports the cluster structure:
  → important predictors for the clustering will have values of $\zeta_j$ close to 1.
- Similarly, use a sparsity inducing prior for $\zeta_j$ with an atom at zero.
Variable Selection: R commands

> profRegr(yModel="Bernoulli",
          xModel="Discrete",
          nSweeps=1000,
          nBurn=1000,
          data="dataMatrix.txt",
          output="output/output",
          nCovariates=5,
          covNames=paste("Var", seq_along(1:10), sep=""),
          nFixedEffects=0,
          xLevels=rep(2,10),
          varSelect="BinaryCluster")
Other features

Other inputs for `profRegr`:

- `hyper` Set hyperparameters
- `nClusInit` Set initial number of clusters, equal to integer
- `sampler` Set sampler: Truncated, SliceDependent or SliceIndependent
- `alpha` Can fix the value of $\alpha$
- `excludeY`
- `extraYVar`
$p(Z|X, Y, \alpha)$ - simulated data
$p(Z|X, Y, \alpha)$ - real data
Further work

- Mixing properties of the algorithm: further moves, tempering
- Extension to include spatial and time components
- New epidemiological and clinical applications and gene-gene and gene-environment interaction search
references