FGSG: Feature Grouping and Selection Over an Undirected Graph

Sen Yang[1,2], Lei Yuan[1,2], Ying-Cheng Lai[3], Xiaotong Shen[4], Peter Wonka[1], Jieping Ye[1,2]
[1]Computer Science and Engineering, [3]Electrical Engineering,
[2]Center for Evolutionary Medicine and Informatics, The Biodesign Institute,
Arizona State University, Tempe, AZ 85287, USA
{senyang, lei.yuan, ying-cheng.lai, peter.wonka, jieping.ye}@asu.edu

[4]School of Statistics, University of Minnesota
Minneapolis, MN 55455, USA
xshen@stat.umn.edu

1 Introduction

FGSG implements algorithms for feature grouping and selection over an undirected graph. It is implemented in C, and compatible with Linux and Windows. FGSG can be directly called in any C function, thus can be interfaced in many environments. It provides the interfaces for Matlab. The linear algebra operations in FGSG are performed by BLAS and LAPACK. You can choose the open-source libraries provided by netlib, or the optimized libraries provided by the computer vendors such as Intel MKL. For Matlab users, the BLAS and LAPACK libraries included in Matlab are recommended[1].

FGSG has 7 feature grouping and selection algorithms:

- Graph Fused Lasso (gflasso).
- Graph OSCAR (goscar).
- Non Convex Feature Grouping and Selection (ncFGS).
- Non Convex Truncated Feature Grouping and Selection (ncTFGS).
- Non Convex Truncated L1 Feature Grouping and Selection (ncTL).
- Non Convex Truncated Fused Feature Grouping and Selection (ncTF).
- Non Convex Truncated L1 and Fused Feature Grouping and Selection (ncTLF).

1.1 How to use FGSG?

The source code is available and you can compile it on your purpose. For Matlab users, you are suggested to first run the Matlab script "mexC" located at the root of the package. Notice that you need to setup C compiler in Matlab before running "mexC"[2]. The package comes also with an example for all functions located at the root of the package.

All functions take the form of

$$\mathbf{x} = \text{functionName}(\mathbf{A}, \mathbf{y}, E, \lambda_1, \lambda_2, \text{opts}).$$

The details of input and output parameters for all functions can be found in Table 1.

---

[1] `http://www.mathworks.com/help/techdoc/matlab_external/br_2m24-1.html`

[2] `http://www.mathworks.com/help/matlab/ref/mex.html`

Table 1: Input and output parameters

Parameter	Description
A	The data matrix Each row corresponds to one sample $(n, p) = size(A)$
y	The response (column vector). The length of y equals to n.
E	the set of edges, $[2, g] = size(E)$
λ_1	The l_1 regularization parameter
λ_2	The grouping penalty parameter
opts	The optional inputs. By default, opts =[ ]. opts.maxIter: the maximum number of iterations in DC programming (default 100) opts.aMaxIter: the maximum number of iterations in ADMM (default 1000) opts.rho: the dual update length for ADMM (default 5). opts.tau: the tuning parameter for non-convex penalty (default 0.15) opts.wt : the weight and signs of edges (default **1**) opts.tol : the tolerance for convergence in DC programming (default 1e-3) opts.aTol : the tolerance for convergence in ADMM (default 1e-3) opts.x0: the initial solution (default 0).
x	The returned weight vector.

2 Functions

2.1 Function gflasso

This function implements graph fused lasso proposed in [1] using ADMM. Given $\mathbf{A} = \{\mathbf{a}_1, \ldots, \mathbf{a}_n\}$, the response $\mathbf{y}$, and a set of edges E, it aims to solves

$$\min \frac{1}{2}||\mathbf{A}\mathbf{x} - \mathbf{y}||^2 + \lambda_1||\mathbf{x}||_1 + \lambda_2 \sum_{(i,j)\in E} w_{(i,j)}|x_i - r_{(i,j)}x_j|,$$

where $w_{(i,j)}$ is the weight of the edge (i, j), and $r_{(i,j)}$ is the sign of the correlation between features $\mathbf{a}_i$ and $\mathbf{a}_j$. The weight and sign can be specified in opts.wt: $\mathbf{w}$ = abs(opts.wt), and $\mathbf{r}$ = sign(opts.wt). If opts.wt is not specified, the default value, a vector of all ones, will be used. Then gflasso solves the following problem:

$$\min \frac{1}{2}||\mathbf{A}\mathbf{x} - \mathbf{y}||^2 + \lambda_1||\mathbf{x}||_1 + \lambda_2 \sum_{(i,j)\in E} |x_i - x_j|.$$

```
% Function x = gflasso(A,y,E,s1,s2,opts)
%      graph fused lasso
%
%% Problem
%
%  min 1/2 ||Ax - y||^2 + s1*||x||_1 + s2*\sum_{(i,j)\in E}w_(i,j)|x_i - r_(i,j)x_j|
%  w_(i,j) is the weight of edge (i,j), r_(i,j) is the sign of correlation between x_i
%  and x_j. w, r can be specified in opts.wt (default value is a vector of 1).
%
%% Input parameters:
%
%  A -               Matrix of size n x p
%  y -               Response vector of size n x 1
%  E -               a set of edges (size of 2 x g)
%  s1-               L_1 norm regularization parameter ( s1 >=0 )
%  s2-               Grouping regularization parameter (s2 >=0)
%  opts -            Optional inputs (defualt value: opts=[]);
%       - aMaxIter   maximum number of iterations in ADMM (default value: 1000)
%       - aTol       tolerance for convergence in ADMM (default value: 1e-3)
%       - x0         initial solution (dimension must be p, default value: opts.x0 = 0)
%       - wt         weight and sign, w = abs(opts.wt), and r = sign(opts.wt)
%                    (dimension must be g)
%       - rho        the dual update length for ADMM (default value: 5)
%
%% Output parameters
%  x -         Solution
%
%  For any problem, please contact with Sen Yang via senyang@asu.edu
%
%  Last modified on June 13, 2012.
%
%% Related papers
%
%  Kim, S. and Xing, E.P., Statistical estimation of correlated genome
%  associations to a quantitative trait network, PLoS genetics, 2009.
```

2.2 Function goscar

The goscar function implements graph OSCAR algorithm proposed in [2] using ADMM. It solves the following optimization problem:

$$\min \frac{1}{2}||\mathbf{A}\mathbf{x}-\mathbf{y}||^2 + \lambda_1||\mathbf{x}||_1 + \lambda_2 \sum_{(i,j)\in E} w_{(i,j)} \max\{|x_i|,|x_j|\}.$$

```
% Function x = goscar(A,y,E,s1,s2,opts)
%      graph OSCAR
%
%% Problem
%
%  min 1/2 ||Ax - y||^2 + s1*||x||_1 + s2*\sum_{(i,j)\in E}w_{(i,j)}max{|x_i|,|x_j|}
%  where w_(i,j) is the weight of edge (i,j), which can be specified in
%  opts.wt (default value is 1)
%% Input parameters:
%
%  A -               Matrix of size n x p
%  y -               Response vector of size n x 1
%  E -               a set of edges (size of 2 x g)
%  s1-               L_1 norm regularization parameter ( s1 >=0 )
%  s2-               Pairwise L_\infty regularization parameter (s2 >=0)
%  opts -            Optional inputs (defualt value: opts=[]);
%       - aMaxIter   maximum number of iterations of ADMM (default value: 1000)
%       - aTol       tolerance for convergence in ADMM (default value: 1e-3)
%       - x0         initial solution (dimension must be p, default value: opts.x0 = 0)
%       - wt         weight w = abs(opts.wt) (dimension must be g)
%       - rho        the dual update length for ADMM (default value: 5)
%
%% Output parameters
%  x -         Solution
%
%  For any problem, please contact with Sen Yang via senyang@asu.edu
%
%  Last modified on June 13, 2012.
%
%% Related papers
%
%  Sen Yang, Lei Yuan, Ying-Cheng Lai, Xiaotong Shen, Peter Wonka, and Jieping Ye,
%  Feature Grouping and Selection over an Undirected Graph, KDD, 2012
```

2.3 Function ncFGS

This function implements non-convex feature grouping and selection algorithm in [2]. It employs ADMM and DC programming to solve

$$\min \frac{1}{2}||\mathbf{A}\mathbf{x}-\mathbf{y}||^2 + \lambda_1||x||_1 + \lambda_2 \sum_{(i,j)\in E} w_{(i,j)}||x_i| - |x_j||.$$

```
% Function x = ncFGS(A,y,E,s1,s2,opts)
%       Non-convex feature grouping and selection
%
%% Problem
%
%  min 1/2 ||Ax - y||^2 + s1*||x||_1 + s2*\sum_{(i,j)\in E}w_{(i,j)}||x_i| - |x_j||
%  where w_(i,j) is the weight of edge (i,j), which can be specified in
%  opts.wt (default value is 1)
%
%% Input parameters:
%
%  A -               Matrix of size n x p
%  y -               Response vector of size n x 1
%  E -               a set of edges (size of 2 x g)
%  s1-               L_1 norm regularization parameter ( s1 >=0 )
%  s2-               Grouping regularization parameter (s2 >=0)
%  opts -            Optional inputs (defualt value: opts=[]);
%       - aMaxIter   maximum number of iterations in ADMM (default value: 1000)
%       - aTol       tolerance of subproblem (ADMM) (default value: 1e-3)
%       - maxIter    maximum number of iterations of DC programming (default value: 100)
%       - tol        tolerance of DC programming (default value: 1e-3)
%       - x0         initial solution (dimension must be p, default value: opts.x0 = 0)
%       - wt         weight w = abs(opts.wt) (dimension must be g)
%       - rho        the dual update length for ADMM (default value: 5)
%
%% Output parameters
%  x -         Solution
%
%  For any problem, please contact with Sen Yang via senyang@asu.edu
%
%  Last modified on June 13, 2012.
%
%% Related papers
%
%  Sen Yang, Lei Yuan, Ying-Cheng Lai, Xiaotong Shen, Peter Wonka, and Jieping Ye,
%  Feature Grouping and Selection over an Undirected Graph, KDD, 2012
```

2.4 Function ncTFGS

This function implements non-convex truncated feature grouping and selection algorithm in [2]. It uses truncated l_1 regularization to reduce estimation bias. It aims at solving

$$min\frac{1}{2}||\mathbf{A}\mathbf{x}-\mathbf{y}||^2 + \lambda_1 p_1(x) + \lambda_2 p_2(x),$$

where

$$\begin{aligned} p_1(x) &= \sum_i (J_\tau(|x_i|)), \\ p_2(x) &= \sum_{(i,j)\in E} w_{(i,j)} J_\tau(||x_i| - |x_j||), \\ J_\tau(x) &= \min(x/\tau, 1). \end{aligned}$$

$J_\tau(x)$ is a surrogate of the l_0 norm. τ is the tuning parameter, specified in opts.tau (0.15 by default).

```
% Function x = ncTFGS(A,y,E,s1,s2,opts)
%       Non-Convex Truncated Feature Grouping and Selection
%
%% Problem
%
%  min 1/2 || Ax - y ||^2 + s1*p_1(x) + s2*p_2(x)
%  where
%    p_1(x) = sum_i(J_\tau(|x_i|))
%    p_2(x) = sum_{(i,j)\in E} w_{(i,j)}J_\tau(||x_i| - |x_j||)
%    J_\tau (x) = min(x/\tau,1) is a surrogate of the L0 norm
%    \tau is tuning parameter for J_\tau, default value 0.15
%    w_(i,j) is the weight of edge (i,j), which can be specified in
%    opts.wt (default value is 1)
%
%% Input parameters:
%
%  A -              Matrix of size n x p
%  y -              Response vector of size n x 1
%  E -              a set of edges (size of 2 x g)
%  s1-              L_1 norm regularization parameter ( s1 >=0 )
%  s2-              Grouping regularization parameter (s2 >=0)
%  opts -           Optional inputs (defualt value: opts=[]);
%       - aMaxIter  maximum number of iterations of ADMM (default value: 1000)
%       - aTol      tolerance for convergence in ADMM (default value: 1e-3)
%       - maxIter   maximum number of iterations of DC programming (default value: 100)
%       - tol       tolerance of DC programming (default value: 1e-3)
%       - x0        initial solution (dimension must be p, default value: opts.x0 = 0)
%       - wt        weight w = abs(opts.wt) (dimension must be g)
%       - rho       the dual update length for ADMM (default value: 5)
%       - tau       the tuning parameter for J_\tau (default 0.15)
%
%% Output parameters
%  x -        Solution
%  funVal-    Function value during iterations
%
%  For any problem, please contact with Sen Yang via senyang@asu.edu
%
%  Last modified on June 13, 2012.
%
%% Related papers
%
%  Sen Yang, Lei Yuan, Ying-Cheng Lai, Xiaotong Shen, Peter Wonka, and Jieping Ye,
%  Feature Grouping and Selection over an Undirected Graph, KDD, 2012
```

2.5 Function ncTL

This function implements a variant of non-convex truncated feature grouping and selection algorithm in [2], which only uses the truncated l_1 regularization. It solves the problem:

$$min\frac{1}{2}||\mathbf{Ax}-\mathbf{y}||^2+\lambda_1\sum_i J_\tau(|x_i|)+\lambda_2\sum_{(i,j)\in E}w_{(i,j)}|x_i-r_{(i,j)}x_j|.$$

```
% Function x = ncTL(A,y,E,s1,s2,opts)
%       Non-Convex Truncated L1 Feature Grouping and Selection
%
%% Problem
%
```

```
%  min 1/2 || Ax - y ||^2 + s1*p_1(x) + s2*p_2(x)
%  where
%    p_1(x) = sum_i(J_\tau(|x_i|))
%    p_2(x) = s2*\sum_{(i,j)\in E}w_(i,j)|x_i - r_(i,j)x_j|
%    J_\tau (x) = min(x/\tau,1) is a surrogate of the L0 norm
%    \tau is tuning parameter for J_\tau, default value 0.15
%    w_(i,j) is the weight of edge (i,j), r_(i,j) is the sign of correlation between x_i
%    and x_j. w, r can be specified in opts.wt (default value is a vector of 1).
%
%% Input parameters:
%
%  A -               Matrix of size n x p
%  y -               Response vector of size n x 1
%  E -               a set of edges (size of 2 x g)
%  s1-               L_1 norm regularization parameter ( s1 >=0 )
%  s2-               Grouping regularization parameter (s2 >=0)
%  opts -            Optional inputs (defualt value: opts=[]);
%       - aMaxIter   maximum number of iterations of ADMM (default value: 1000)
%       - aTol       tolerance for convergence in ADMM (default value: 1e-3)
%       - maxIter    maximum number of iterations of DC programming (default value: 100)
%       - tol        tolerance of DC programming (default value: 1e-3)
%       - x0         initial solution (dimension must be p, default value: opts.x0 = 0)
%       - wt         weight and sign, w = abs(opts.wt), and r = sign(opts.wt) (dimension
%                    must be g)
%       - rho        the dual update length for ADMM (default value: 5)
%       - tau        the tuning parameter for J_\tau (default 0.15)
%
%% Output parameters
%  x -         Solution
%  funVal-     Function value during iterations
%
%  For any problem, please contact with Sen Yang via senyang@asu.edu
%
%  Last modified on June 13, 2012.
%
%% Related papers
%
%  Sen Yang, Lei Yuan, Ying-Cheng Lai, Xiaotong Shen, Peter Wonka, and Jieping Ye,
%  Feature Grouping and Selection over an Undirected Graph, KDD, 2012
```

2.6 Function ncTF

This function implements a variant of non-convex truncated feature grouping and selection algorithm in [2], which only uses the truncated fused penalty. It aims to solving the following problem:

$$min\frac{1}{2}||\mathbf{Ax}-\mathbf{y}||^2+\lambda_1\|\mathbf{x}\|_1+\lambda_2\sum_{(i,j)\in E}w_{(i,j)}J_\tau(|x_i-r_{(i,j)}x_j|).$$

```
% Function x = ncTF(A,y,E,s1,s2,opts)
%       Non-Convex Truncated Fused Feature Grouping and Selection
%
%% Problem
%
%  min 1/2 || Ax - y ||^2 + s1*p_1(x) + s2*p_2(x)
%  where
%    p_1(x) = sum_i(|x_i|)
%    p_2(x) = s2*\sum_{(i,j)\in E}w_(i,j)J_\tau(|x_i - r_(i,j)x_j|)
%    J_\tau (x) = min(x/\tau,1) is a surrogate of the L0 norm
%    \tau is tuning parameter for J_\tau, default value 0.15
%    w_(i,j) is the weight of edge (i,j), r_(i,j) is the sign of correlation between x_i
%    and x_j. w, r can be specified in opts.wt (default value is a vector of 1).
%
%% Input parameters:
%
%  A -                Matrix of size n x p
%  y -                Response vector of size n x 1
%  E -                a set of edges (size of 2 x g)
%  s1-                L_1 norm regularization parameter ( s1 >=0 )
%  s2-                Grouping regularization parameter (s2 >=0)
%  opts -             Optional inputs (defualt value: opts=[]);
%       - aMaxIter    maximum number of iterations of ADMM (default value: 1000)
%       - aTol        tolerance for convergence in ADMM (default value: 1e-3)
%       - maxIter     maximum number of iterations of DC programming (default value: 100)
%       - tol         tolerance of DC programming (default value: 1e-3)
%       - x0          initial solution (dimension must be p, default value: opts.x0 = 0)
%       - wt          weight and sign, w = abs(opts.wt), and r = sign(opts.wt) (dimension
%                     must be g)
%       - rho         the dual update length for ADMM (default value: 5)
%       - tau         the tuning parameter for J_\tau (default 0.15)
%
%% Output parameters
%  x -          Solution
%  funVal-      Function value during iterations
%
%  For any problem, please contact with Sen Yang via senyang@asu.edu
%
%  Last modified on June 13, 2012.
%
%% Related papers
%
%  Sen Yang, Lei Yuan, Ying-Cheng Lai, Xiaotong Shen, Peter Wonka, and Jieping Ye,
%  Feature Grouping and Selection over an Undirected Graph, KDD, 2012
```

2.7 Function ncTLF

This function implements a variant of non-convex truncated feature grouping and selection algorithm in [2], which uses both the truncated l_1 regularization and the truncated fused penalty. It aims to solving the following problem:

$$min\frac{1}{2}||\mathbf{Ax}-\mathbf{y}||^2+\lambda_1\sum_i J_\tau(|x_i|)+\lambda_2\sum_{(i,j)\in E} w_{(i,j)}J_\tau(|x_i-r_{(i,j)}x_j|).$$

```
% Function x = ncTLF(A,y,E,s1,s2,opts)
%       Non-Convex Truncated L1 and Fused Feature Grouping and Selection
%
%% Problem
%
%  min 1/2 || Ax - y ||^2 + s1*p_1(x) + s2*p_2(x)
%  where
%    p_1(x) = sum_i(J_\tau(|x_i|))
%    p_2(x) = s2*\sum_{(i,j)\in E}w_(i,j)J_\tau(|x_i - r_(i,j)x_j|)
%    J_\tau (x) = min(x/\tau,1) is a surrogate of the L0 norm
%    \tau is tuning parameter for J_\tau, default value 0.15
%    w_(i,j) is the weight of edge (i,j), r_(i,j) is the sign of correlation between x_i
%    and x_j. w, r can be specified in opts.wt (default value is a vector of 1).
%
%% Input parameters:
%
```

```
%  A -               Matrix of size n x p
%  y -               Response vector of size n x 1
%  E -               a set of edges (size of 2 x g)
%  s1-               L_1 norm regularization parameter ( s1 >=0 )
%  s2-               Grouping regularization parameter (s2 >=0)
%  opts -            Optional inputs (defualt value: opts=[]);
%       - aMaxIter   maximum number of iterations of ADMM (default value: 1000)
%       - aTol       tolerance for convergence in ADMM (default value: 1e-3))
%       - maxIter    maximum number of iterations of DC programming (default value: 100)
%       - tol        tolerance of DC programming (default value: 1e-3)
%       - x0         initial solution (dimension must be p, default value: opts.x0 = 0)
%       - wt         weight and sign, w = abs(opts.wt), and r = sign(opts.wt) (dimension
%                    must be g)
%       - rho        the dual update length for ADMM (default value: 5)
%       - tau        the tuning parameter for J_\tau (default 0.15)
%
%% Output parameters
%  x -         Solution
%  funVal-     Function value during iterations
%
%  For any problem, please contact with Sen Yang via senyang@asu.edu
%
%  Last modified on June 13, 2012.
%
%% Related papers
%
%  Sen Yang, Lei Yuan, Ying-Cheng Lai, Xiaotong Shen, Peter Wonka, and Jieping Ye,
%  Feature Grouping and Selection over an Undirected Graph, KDD, 2012
```

References

[1] S. Kim and E. Xing. Statistical estimation of correlated genome associations to a quantitative trait network. *PLoS genetics*, 5(8):e1000587, 2009.

[2] S. Yang, L. Yuan, Y. Lai, X. Shen, P. Wonka, and J. Ye. Feature grouping and selection over an undirected graph. *KDD*, 2012.