regsem: Regularized Structural Equation Modeling

Ross Jacobucci

University of Notre Dame

Abstract

The **regsem** package in R, an implementation of regularized structural equation modeling (RegSEM; Jacobucci, Grimm, and McArdle 2016), was recently developed with the goal incorporating various forms of penalized likelihood estimation with a broad array of structural equations models. The forms of regularization include both the *ridge* (Hoerl and Kennard 1970) and the least absolute shrinkage and selection operator (*lasso*; Tibshirani 1996), along with sparser extensions. RegSEM is particularly useful for structural equation models that have a small parameter to sample size ratio, as the addition of penalties can reduce the complexity, thus reducing the bias of the parameter estimates. The paper details the algorithmic details and an overview of the use of **regsem** with the application of both factor analysis and latent growth curve models.

Keywords: regularization, structural equation modeling, latent variables, R.

1. Introduction

The desire for simplicity in model structure comes by many names, including simple structure (Thurstone 1935), variable complexity (Browne 2001), parsimony (Raykov and Marcoulides 1999; Marsh and Hau 1996), "sparse loadings" in the context of principal components analysis (Zou, Hastie, and Tibshirani 2006), and lastly, "sparsistency", denoting that all parameters in a sparse model that are zero are correctly estimated as zero with probability tending to one (Lam and Fan 2009). The goal is to accurately and efficiently estimate a model that is parsimonious in allowing users to easily interpret the model's representation of reality. In the context of latent variables, reducing the complexity of models can come in many forms: selecting among multiple predictors of a latent variable, simplifying factor structure by removing cross-loadings, determining whether the addition of nonlinear terms are necessary in longitudinal models, and many others. As a simple running example, Figure 1 depicts a linear latent growth curve model (e.g. Meredith and Tisak 1990) with four time points and ten predictors for a simulated dataset.
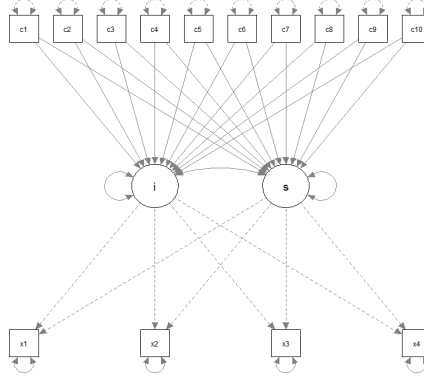
Figure 1: Growth curve model with 10 predictors of both the intercept and slope

As an example, a researcher may want to test this model, but may only have a relatively small sample size (e.g. 80). However, there are 29 estimated parameters in this model, resulting in a estimated parameter to sample size ratio as far below even the most liberal recommendations (e.g. 10:1 parameters to sample size; Kline 2015). In lieu of finding additional respondents, reducing the number of parameters estimated is one effective strategy for reducing bias. Specifically, the 20 estimated regressions from *c1-c10* could be reduced to a number that makes the ratio of the parameters estimated to sample size more reasonable. To explore this further, the next section provides an overview of regularization, and how different forms can be used to perform variable selection across a broad range of models.

2. Regularization

Although a whole host of methods exist to perform variable selection, the use of regularization has seen a wide array of application in the context of regression, and more recently, in areas such as graphical modeling, as well as a host of others. The two most common procedures for regularization in regression are the *ridge* (Hoerl and Kennard 1970) and the least absolute shrinkage and selection operator (*lasso*; Tibshirani 1996); however, there are various alternative forms that can be seen as subsets or generalizations of these two procedures. Given an outcome vector y and predictor matrix $X \in R^{n \times p}$, the ridge estimates are defined as

$$\hat{\beta}^{ridge} = argmin\Big\{ \sum_{i=1}^{N}(y_i = \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \Big\}, \tag{1}$$

where β_0 is the intercept, β_j is the coefficient for x_j, and λ is the penalty that controls the amount of shrinkage. Note that when $\lambda = 0$, Equation 3 reduces to ordinary least squares regression. As λ is increased, the β parameters are shrunken towards zero. The lasso estimates are defined as

$$\hat{\beta}^{lasso} = argmin\Big\{ \sum_{i=1}^{N}(y_i = \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \Big\}. \tag{2}$$

In lasso regression, the l_1-norm is used, instead of l_2-norm as in ridge, which also shrinks the β parameters, but additionally drives the parameters all the way to zero, thus performing a form of subset selection.

In the context of our example depicted in Figure 1, to use lasso regression to select among the covariates, the growth model would need to be reduced to two factor scores, which neglects both the relationship between both the slope and intercept, reducing both to independent factor scores. Particularly in models with a greater number of latent variables, this becomes increasingly problematic. A method that keeps the model structure, while allowing for penalized estimation of specific parameters is regularized structural equation modeling (RegSEM; Jacobucci, Grimm, and McArdle 2016). RegSEM adds a penalty function to the traditional maximum likelihood estimation (MLE) for structural equation models (SEMs). The maximum likelihood cost function for SEMs can be written as

$$F_{ML} = log(|\Sigma|) + tr(C * \Sigma^{-1}) - log(|C|) - p. \tag{3}$$

where Σ is the model implied covariance matrix, C is the observed covariance matrix, and p is the number of estimated parameters. RegSEM builds in an additional element to penalize certain model parameters yielding

$$F_{regsem} = F_{ML} + \lambda P(\cdot) \tag{4}$$

where λ is the regularization parameter and takes on a value between zero and infinity. When λ is zero, MLE is performed, and when λ is infinity, all penalized parameters are shrunk to zero. $P(\cdot)$ is a general function for summing the values of one or more of the model's parameter matrices. The two common forms of $P(\cdot)$ include both the lasso ($\| \cdot \|_1$), which penalizes the sum of the absolute values of the parameters, and ridge ($\| \cdot \|_2$), which penalizes the sum of the squared values of the parameters.

In our example, the twenty regression parameters from the covariates to both the intercept and slope would be penalized. Using lasso penalties, the absolute value of these twenty parameters would be summed and after being multiplied by the penalty λ, added to equation 4, resulting in:

$$F_{lasso} = F_{ML} + \lambda * \left\| \begin{array}{c} c1 \rightarrow i \\ c2 \rightarrow i \\ \vdots \\ c10 \rightarrow i \\ c1 \rightarrow s \\ \vdots \\ c10 \rightarrow s \end{array} \right\|_1 \tag{5}$$

Although the fit of the model is easily calculated given a set of parameter estimates, traditional optimization procedures for SEM cannot be used given the non-differentiable nature of lasso penalties, and as detailed later, sparse extensions.

2.1. Optimization

One method that has become popular for optimizing penalized likelihood method is that of proximal gradient descent (e.g. p. 104 in Hastie, Tibshirani, and Wainwright 2015). In comparison to one-step procedures common in SEM optimization, that only involve a method for calculating the step size and the direction (typically using the gradient and an approximation of the Hessian), proximal gradient descent can be formulated as a two-step procedure. With a stepsize of s^t and parameters θ^t at iteration t:

1. First, take a gradient step size $z = \theta^t - s^t \nabla g(\theta^t)$.

2. Second, perform elementwise soft-thresholding $\theta^{t+1} = S_{s^t \lambda}(z)$.

where $S_{s^t \lambda}(z)$ is the soft-thresholding operator (Donoho 1995) used to overcome non-differentiability of the lasso penalty at the origin:

$$S_{s^t \lambda}(z_j) = sign(z_j)(|z_j| - s^t \lambda)_+ \tag{1}$$

where $(x)_+$ is shorthand for max(x,0) and s^t is the step size. Henceforth, λ is assumed to encompass both the penalty and the step size s^t. This procedure is only used to update parameters that are subject to penalty. Non-penalized parameters are updated only using step 1 from above.

In contrast to only updating one parameter at a time in a coordinate-wise fashion, for RegSEM the optimization steps can be divided by both the A and S matrices. This block-wise gradient descent manifests itself as:

Algorithm 1 RegSEM Block Coordinate Descent

1: Generate starting values for θ_t
2: Calculate initial fit F_t
3: Set step size s. .1 works well at this time.
4: Set tolerance (tol). e.g. 1e-6
5: **while** $|F_t - F_{t+1}| > tol$ **do**
6:     Calculate gradient for A:$\nabla(A) =: \frac{\partial A}{\partial \theta_t}$
7:     $\theta_{t+1*,A} =: \theta_{t,A} - s\nabla(A)$
8:     Update penalized parameters: $\theta_{t+1*,A(pen)} =: S_{s\lambda}(\theta_{t+1*,A(pen)})$
9:     Calculate gradient for S:$\nabla(S) =: \frac{\partial S}{\partial \theta_{t+1*}}$
10:    Update S parameters: $\theta_{t+1*,S} =: \theta_{t,S} - s\nabla(S)$
11:    $\theta_{t+1} =: \theta_{t+1*}$
12:    Update S_{t+1}, A_{t+1}
13:    $\Sigma_{t+1} = F(I - A_{t+1})^{-1} S_{t+1} (I - A_{t+1})^{-T} F^T$
14:    $F_{t+1} = F_{ML}(\Sigma_{t+1}, C) + \lambda \|A_{t+1}(pen)\|$

---

where step 13 is the calculation of the implied covariance matrix using RAM matrices and step 14 calculates the fit of the model with penalizing parameters in the A matrix. Note that parameters from the S matrix can also be penalized, however, this is much less common. This algorithm can be described as first order proximal block coordinate descent. For some SEM models, using block updates has been found to work better than standard gradient descent with the same soft thresholding of penalized parameters.

3. Types of Penalties

Outside of both ridge and lasso penalties, a whole host of additional forms of regularization exist.

3.1. Elastic Net

Most notably, the elastic net (Zou and Hastie 2005) encompasses both the ridge and lasso, reaching a compromise between both through the addition of an additional parameter α, manifesting itself as

$$P_{enet}(\theta_j) = 0.5(1 - \alpha)\|\theta_j\|_2 + \alpha\|\theta_j\|_2$$

with a soft-thresholding update of

$$S(\theta_j) = \begin{cases} 0, & |\theta_j| < \alpha\lambda \\ \frac{sgn(\theta_j)(|\theta_j| - \alpha\lambda)}{1 + (1 - \alpha)\lambda}, & |\theta_j| \geq \alpha\lambda \end{cases}$$

When α is zero, ridge is performed, and conversely when α is 1, lasso regularization is performed. This method harnesses the benefits of both methods, particularly when variable selection is warranted (lasso), but there may be collinearity between the variables (ridge).

3.2. Adaptive Lasso

In using lasso penalties, difficulties emerge when the scale of variables differ dramatically. By only using one value of λ, this can add appreciable bias to the resulting estimates (e.g. Fan and Li 2001). One method proposed for overcoming this limitation is the adaptive lasso (Zou 2006). Instead of penalizing parameters directly, each parameter is scaled by the un-penalized estimated (MLE parameter estimates in SEM). The adaptive lasso results in:

$$F_{alasso} = F_{ML} + \lambda\|\theta_{ML}^{-1} * \theta_{pen}\|_1$$

with, following the same form for the lasso, the soft-thresholding update is:

$$S(\theta_j) = sign(\theta_j)(|\theta_j| - \frac{\lambda}{2|\theta_j|})_+$$

In this, larger penalties are given for non-significant (smaller) parameters, limiting the bias in estimating larger, significant parameters. Note that one limitation of this approach for SEM
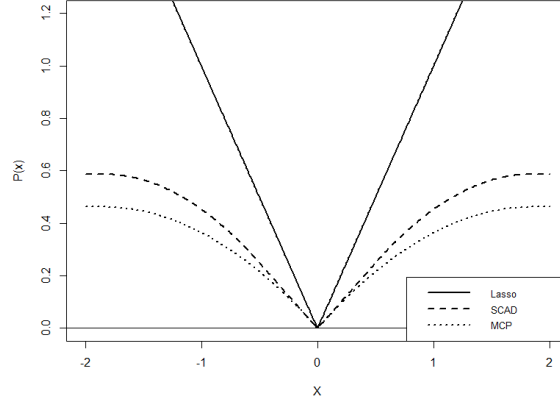
Figure 2: Comparison of types of penalties with $\lambda = 0.5$

models is that the model needs to be estimable with MLE. Particularly for models with large numbers of variables, in relation to sample size, this may not be possible.

3.3. Smoothly Clipped Absolute Deviation Penalty

Two additional penalties that overcome some of the deficiencies of the lasso, producing sparser solutions, include the smoothly clipped absolute deviation penalty (SCAD; Fan and Li 2001) and the minimax concave penalty (MCP; Zhang 2010). In comparison to the lasso, both the SCAD and MCP have much smaller penalties for large parameters, where the amount of penalty for small penalties is similar to the lasso, as is evident in Figure 2.

The SCAD takes the form of:

$$pen_{\lambda,\gamma}(\theta_j) = \lambda\big\{I(\theta_j \leq \lambda) + \frac{(\gamma\lambda - 0)_+}{(\gamma - 1)\lambda}I(\theta_j > \lambda)\big\}$$

with a soft-thresholding update of

$$S(\theta_j) = \begin{cases} S(\theta_j, \lambda), & |\theta_j| \geq 2\lambda \\ \frac{\gamma-1}{\gamma-2}S(\theta_j, \frac{\lambda\gamma}{\gamma-1}), & 2\lambda < |\theta_j| \leq \alpha\lambda \\ \theta_j & |\theta_j| > \lambda\gamma \end{cases}$$

for $\gamma > 2$. As the the penalty in equation 11 is non-convex (as is the MCP), this makes the computation more difficult. However, in the context of SEM this can be seen as less problematic, as equation 3 is also non-convex.

3.4. Minimax Concave Penalty

Zhang (2010) proposed the minimax concave penalty:

$$pen_{\lambda,\gamma}(\theta_j) = \lambda\left(|\theta_j| - \frac{\theta_j^2}{2\lambda\gamma}\right)I(|\theta_j| < \lambda\gamma) + \frac{\lambda^2\gamma}{2}I(|\theta_j| \geq \lambda\gamma)$$

with a soft-thresholding update of

$$S(\theta_j) = \begin{cases} \frac{\gamma}{\gamma-1}S(\theta_j, \lambda), & |\theta_j| \leq \lambda\gamma \\ \theta_j & |\theta_j| > \lambda\gamma \end{cases}$$

for $\gamma > 0$. As seen in Figure 2, this results in similar amount of shrinkage for smaller estimates in comparison to the SCAD, however, less for larger estimates. For both the SCAD and MCP, both the γ and λ parameters are used as hyper-parameters. This involves testing models over a two-dimensional array of parameters, however, γ can be fixed to 3.7 per Fan and Li (2001).

4. Implementation

RegSEM is implemented as the **regsem** package (Jacobucci 2017) in the R statistical environment (R Core Team 2017). To estimate the maximum likelihood fit of the model, **regsem** uses *Reticular Action Model* (RAM; J. J. McArdle and McDonald 1984, McArdle (2005)) notation to derive an implied covariance matrix. The parameters of each SEM are translated into three matrices: the *filter* (F), the *asymmetric* (A; directed paths; e.g. factor loadings or regressions), and the *symmetric* (S; undirected paths; e.g. covariances or variances). See Jacobucci, Grimm, and McArdle (2016) for more detail on RAM notation.

Syntax for using the **regsem** is based on the **lavaan** package (Rosseel 2012) for structural equation models. **lavaan** is a general SEM software program that can fit a wide array of models with various estimation methods. To use **regsem**, the user has to first fit the model in **lavaan**. Note that particularly in cases that the number of variables is larger than the sample size, the model in lavaan does not need to converge, let alone run. In this case, the `do.fit=FALSE` argument in lavaan can be used. ###maybe move previous sentences### Below is the code for a one factor model fit in lavaan with the Holzinger and Swineford (1939) dataset.

```
library(lavaan)
mod <- "
f1 = ~ NA*x1 + x2 + x3 + x4 + x5 + x6 + x7 + x8 + x9
f1~~1*f1
"
out <- cfa(mod,HolzingerSwineford1939)
```

After a model is run in lavaan, using `lavaan()` or any of the wrapper functions for fitting a model (i.e. `sem()`, `cfa()`, or `growth()`), the object is then used by the regsem package to translate the model into RAM notation and run using one of three functions: `regsem()`, `multi_optim()`, or `cv_regsem()`. The `regsem()` function runs a model with one penalty value, whereas `multi_optim()` does the same but allows for the use of random starting values. However, the main function is `cv_regsem()` as this not only runs the model, but runs

it across a vector of varying penalty values. For instance in the above one-factor model, each of the factor loadings can be tested with lasso penalties to determine whether each indicator is a necessary component of the latent factor:

```
library(regsem)
extractMatrices(out)["A"]
out.reg <- cv_regsem(out, type="lasso",
                     pars_pen=c(1:9),n.lambda=15,jump=.05)
```

In this, the function `extractMatrices()` allows the user to examine at how the lavaan model is translated into RAM matrices. Further, by looking at the A matrix, the parameter numbers corresponding to the factor loadings of interest for regularization can be identified. For this model, the factor loadings represent parameter numbers one through nine, of which we pass directly to the `pars_pen` argument of the `cv_regsem()` function (if `pars_pen=NULL` then all directed effects, outside of intercepts, are penalized). Additionally, we pass the arguments of how many values of penalty we want to test (`n.lambda=15`), how much the penalty should increase for each model (`jump=.05`), and finally that lasso estimation is used (`type="lasso"`).

The `out.reg` object contains two components, `out.reg[[1]]` has the parameter estimates for each of the 15 models,

```
head(round(out.reg[[1]],2),5)
```

```
     f1 -> x1 f1 -> x2 f1 -> x3 f1 -> x4 f1 -> x5 f1 -> x6 f1 -> x7
[1,]     0.51     0.26     0.25     0.98     1.08     0.92     0.20
[2,]     0.41     0.17     0.17     0.89     0.98     0.83     0.12
[3,]     0.33     0.10     0.10     0.83     0.91     0.77     0.06
[4,]     0.27     0.04     0.04     0.78     0.87     0.72     0.01
[5,]     0.21     0.00     0.00     0.75     0.83     0.69     0.00
     f1 -> x8 f1 -> x9 x1 ~~ x1 x2 ~~ x2 x3 ~~ x3 x4 ~~ x4 x5 ~~ x5
[1,]     0.20     0.31     1.10     1.31     1.21     0.38     0.49
[2,]     0.13     0.23     1.11     1.32     1.22     0.38     0.48
[3,]     0.08     0.17     1.13     1.33     1.23     0.38     0.47
[4,]     0.03     0.12     1.15     1.35     1.25     0.38     0.46
[5,]     0.00     0.07     1.18     1.36     1.26     0.39     0.46
     x6 ~~ x6 x7 ~~ x7 x8 ~~ x8 x9 ~~ x9
[1,]     0.36     1.15     0.98     0.92
[2,]     0.36     1.15     0.99     0.93
[3,]     0.36     1.16     1.00     0.94
[4,]     0.37     1.17     1.01     0.96
[5,]     0.38     1.18     1.02     0.97
```

while `out.reg[[2]]` looks like:

```
round(out.reg[[2]],2)
```

```
      lambda conv rmsea     BIC
 [1,]   0.00    0  0.19 7805.18
 [2,]   0.05    0  0.19 7815.39
 [3,]   0.10    0  0.20 7840.59
 [4,]   0.15    0  0.21 7875.84
 [5,]   0.20    0  0.20 7886.15
 [6,]   0.25    0  0.21 7911.82
 [7,]   0.30    0  0.21 7934.45
 [8,]   0.35    0  0.22 7960.67
 [9,]   0.40    0  0.23 7991.03
[10,]   0.45    0  0.23 8009.42
[11,]   0.50    0  0.23 8034.81
[12,]   0.55    0  0.24 8069.49
[13,]   0.60    0  0.29 8360.54
[14,]   0.65    0  0.29 8360.53
[15,]   0.70    0  0.29 8360.52
```

In this, the user can examine the penalty (lambda), whether the model converged ("conv"=0), and the fit of each model. By default, two fit indices are output, both the root mean square error of approximation (RMSEA; Steiger and Lind 1980), and the Bayesian information criteria (BIC; Schwarz 1978). Both the RMSEA and BIC take into account the degrees of freedom of the model, an important point for model selection in the presence of lasso penalties (and other penalties that set parameters to zero). Based on work done on degrees of freedom for lasso regression, as Zou, Hastie, and Tibshirani (2007) proved that the number of nonzero coefficients is an unbiased estimate of the degrees of freedom for regression. As the penalty increases, select parameter are set to zero, thus the degrees of freedom increases, which for fit indices that include the degrees of freedom in the calculation, means that although the likelihood of the model may only get worse (increase), both the RMSEA and BIC can improve (decrease) due to the impact of increased degrees of freedom.

Instead of examining the `out.reg[[1]]` output matrix of parameter estimates, users also have the option to plot the trajectory of each of the penalized parameters. This is accomplished with `plot_cv(out.reg,pars=1:9)`, resulting in:

```
plot_cv(out.reg,pars=1:9)
```
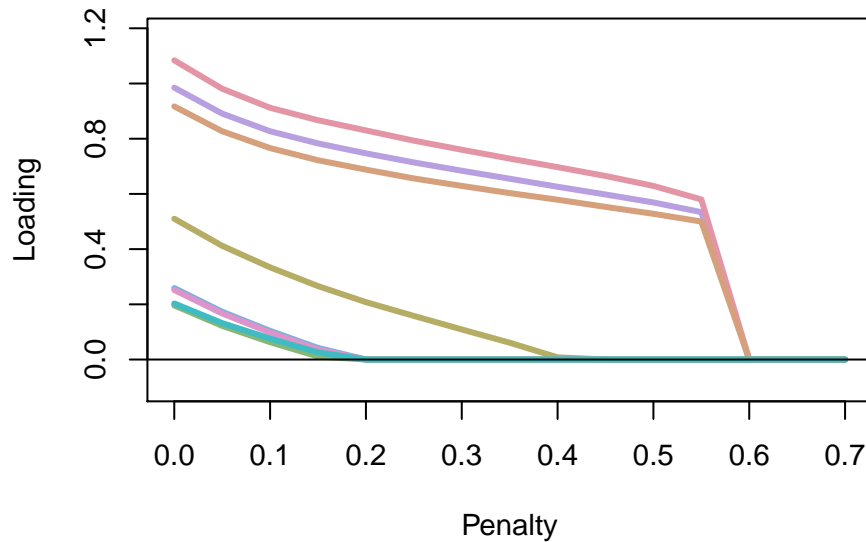
After a final model (penalty) is chosen, users have the option either just use the output from `cv_regsem()`, or the final model can be re-run with either `regsem()` or `multi_optim()` to attain additional information. In the model above, the best fitting penalty, according to the BIC, is $\lambda = 0$. However, for demonstration purposes, we can choose a penalty of 0.2. With this, the model is re-run with `regsem()`

```
mod.out <- regsem(out, type="lasso", pars_pen=c(1:9),lambda=0.2)
#summary(mod.out)
```

Note that the arguments used above correspond to the same arguments in `multi_optim()`. However, `multi_optim()` has additional optional arguments corresponding to the number or random starts to try. Additional fit indices can be attained through the `fit_indices()` function.

```
fit_indices(mod.out)
```

```
$Data_Type
[1] "Train"

$fits
        Fmin          varFit              p          chisq        p.chisq
     0.69087         0.00000        9.00000      415.90286        0.00000
        nfac              df           npar              N  baseline.chisq
     1.00000        31.00000       14.00000      301.00000      918.85159
  baseline.df            logl            ncp           rmsea     rmsea.lower
```

36.00000	-3903.04360	1.28301	0.20344	0.18598
rmsea.upper	rmsea.pval	CFI	TLI	BIC
NA	0.00000	0.56402	0.49370	7885.98674
AIC	CAIC	EBIC.5	EBIC.25	
7834.08719	7983.11043	8051.12157	8017.06459	

These same fit measures can be accessed through `cv_regsem`, through changing the defaults with the `fit.ret=c("rmsea","BIC")` argument. Finally, instead of assessing these fit indices on the same sample that the models were run on, a holdout dataset could be used. This can be done two ways: either with `cv_regsem(...,fit.ret2="test")` or with `fit_indices(model,CV=TRUE,CovMat=)` and specifying the name of the holdout covariance matrix.

5. Comparison

To compare the different types of penalties in **regsem**, we return to the the initial example of the latent growth curve model displayed in Figure 1. Using the same simulated data, the model can be run in **lavaan** as

```
mod1 <- "
i =~ 1*x1 + 1*x2 + 1*x3 + 1*x4
s =~ 0*x1 + 1*x2 + 2*x3 + 3*x4
i ~ c1 + c2 + c3 + c4 + c5 + c6 + c7 + c8 + c9 + c10
s ~ c1 + c2 + c3 + c4 + c5 + c6 + c7 + c8 + c9 + c10
"
lav.growth <- growth(mod1,dat,fixed.x=T)
```

To compare different types of penalties in **regsem**, it requires a different specification of the `type` argument. The options currently include maximum likelihood (`"none"`), ridge (`"ridge"`), lasso (`"lasso"`), adaptive lasso (`"alasso"`), elastic net (`"enet"`), SCAD (`"scad"`), and MCP (`"mcp"`). For the elastic net, there is an additional hyperparameter, α that controls the tradeoff between ridge and lasso penalties. This is specified as `alpha=` , which has a default of 0.5. Additionally, both the SCAD and MCP have the additional hyper parameter of γ, which is specified as `gamma=` and defaults to 3.7 per Fan and Li (2001).

For the purposes of comparison, each of the 20 covariate regressions were penalized using the lasso, adaptive lasso, SCAD, and MCP, and compared to the maximum likelihood estimates. In this model, the data were simulated to have two large effects (both *c1* parameters), two small effects (both *c2* parameters) and sixteen true zero effects (*c3-c10* parameters). Note that the covariates were simulated to have zero covariance among each variable. If there was substantial collinearity among covariates, the elastic net would be more appropriate to simultaneously select predictors while also accounting for the collinearity. The parameter estimates corresponding the the best fit of the BIC are has the fit of each model, resulting in Table 1, created using the **xtable** package (Dahl 2009).

While every regularization method erroneously set both simulated true intercept effects as zero (as in maximum likelihood, which accounts for these errors), both the adaptive lasso

	ML	lasso	alasso	SCAD	MCP
c1 -> i	0.92*	0.72	0.91	0.94	0.92
c2 -> i	0.07	0.00	0.00	0.00	0.00
c3 -> i	0.10	0.00	0.00	0.00	0.00
c4 -> i	0.07	0.00	0.00	0.00	0.00
c5 -> i	0.04	0.00	0.00	0.00	0.00
c6 -> i	-0.25	0.00	0.00	0.00	-0.19
c7 -> i	0.11	0.00	0.00	0.00	0.00
c8 -> i	-0.13	0.00	0.00	0.00	0.00
c9 -> i	-0.03	0.00	0.00	0.00	0.00
c10 -> i	0.09	0.00	0.00	0.00	0.00
c1 -> s	1.18*	1.09	1.22	1.24	1.24
c2 -> s	0.29*	0.19	0.28	0.35	0.35
c3 -> s	0.18	0.09	0.00	0.00	0.00
c4 -> s	-0.08	0.00	0.00	0.00	0.00
c5 -> s	-0.18	0.00	0.00	0.00	0.00
c6 -> s	0.25*	0.00	0.00	0.00	0.00
c7 -> s	-0.18	-0.04	0.00	0.00	0.00
c8 -> s	0.26*	0.10	0.00	0.00	0.00
c9 -> s	-0.06	0.00	0.00	0.00	0.00
c10 -> s	0.08	0.00	0.00	0.00	0.00
BIC	3465.28	3427.46	3415.05	3414.38	3417.20

Table 1: Parameter estimates for the final models across five estimation methods. Note that * represent significant parameters at p < .05 for maximum likelihood estimation.

and SCAD correctly identified every true zero effect. The lasso identified two false effects while the MCP mistakenly identified one. Additionally, the lasso estimation of the true effects was attentuated in comparison to the other regularization methods. This is in line with previous research (Fan and Li 2001), necessitating the use of a two-step relaxed lasso method (Meinshausen 2007, see Jacobucci, Grimm, and McArdle (2016)) As expected given the small ratio between number of estimated parameters and sample size, maximum likelihood mistakenly identified 3 false effects as significant.

To compare the performance of each penalization method further, particularly in the presence of small number of parameters to sample size ratios, a small scale simulation study was conducted. The same model and effects was kept, but the sample size was varied to include 80, 200, and 1000 to demonstrate how maximum likelihood estimation of effects improves as sample size increases, while each of the regularization methods performs well regardless of sample size. Each run was replicated 200 times. For each regularization method, the BIC was used to choose a final model among the 40 penalty vales. The results are displayed in Table 2.

For false positive, the adaptive lasso demonstrated the best performance, where the performance of maximum likelihood estimation leveled off at the 0.05 level at a sample size of 1000 as expected. For false negatives, lasso penalties demonstrated similar results to maximum likelihood. This was expected given the tendency of the lasso to under-penalize small coefficients in comparison to the other regularization methods. The adaptive lasso and

	N	ML	lasso	alasso	SCAD	MCP
	80.00	0.08	0.08	**0.04**	0.05	**0.20**
False Positives	200.00	0.06	0.05	**0.02**	0.03	0.06
	1000.00	0.05	0.08	**0.01**	**0.01**	0.02
	80.00	0.33	**0.31**	0.35	0.35	0.31
False Negatives	200.00	**0.19**	**0.19**	0.23	0.26	0.27
	1000.00	**0.00**	**0.00**	**0.00**	0.01	0.03

Table 2: Results from the simulation using the model in Figure 1. Each condition was replicated 200 times. False positives represent concluding that the simulated regressions of zero were concluded as nonzero. False negatives are concluding that either the simulated regression values of 1 or 0.2 are in fact zero. Bolded values represent the smallest error per condition.

SCAD demonstrated slightly worse results, however, outside of the MCP, each method made either zero or near zero errors at a sample size of 1000. The poor performance of the MCP may be in part due to fixing the γ penalty to 3.7. Varying this parameter may improve the performance of the method. In summary, the regularization methods demonstrated an improvement over maximum likelihood, particularly at small samples, for a model that had a large number of estimated parameters.

6. Discussion

This paper provides an introduction to the **regsem** package, outlining the mathematical details of the regularized structural equation modeling (RegSEM; Jacobucci, Grimm, and McArdle 2016) method and the usage of the **regsem** package. RegSEM allows the use of regularization while keeping the structural equation model intact, adding penalization directly into the estimation of the model. The application of RegSEM was detailed using two example models: a latent growth curve model with 20 predictors of both the latent intercept and slope, along with a factor analysis with one latent factor. With the latent growth curve model, the small parameter to sample size ratio resulted in a larger number of false positives in using maximum likelihood estimation. In both the simulated example and the small simulation, the different types of regularization in **regsem** demonstrated better false positive and negative rates in comparison to maxiumum likelihood across sample sizes.

Broadly speaking, there is a growing amount of research into the integration between data mining methods and latent variable models. Specifically, beyond RegSEM, this has taken the form of item response theory and regularization (Sun et al. 2016), pairing both structural equation models with decision trees (Brandmaier et al. 2013), exploratory psychological network analaysis (e.g. Epskamp, Rhemtulla, and Borsboom 2016), along with many others. The amount of pairing between methods that have generally been housed in separate camps will only increase into the future. This type of research will be facilitated by the general upsurge in the creation of open source software that gives users a general framework to test models. This was the motivation behind creating the **regsem** package in that users can estimate models ranging from simple factor analysis models, to latent longitudinal models with few to many time points, and finally to models with a large number of latent and observed variables. The use of regularization allows for the estimation of much larger structural equa-

tion models than before. Specifically, given that latent variables, as discussed in this article, are more commonly used in the social and behavioral sciences, sample sizes are generally not large. To estimate large models with small sample sizes invites increasing amounts of bias, and as demonstrated with the simulated data in this paper. Regularization can be used to reduce the complexity of the model, thus decreasing both the bias and variance.

With highly constrained structural equation models, achieving model convergence can be particularly problematic in using **regsem**. For instance, with the latent change score model (McArdle and Hamagami 2001), Bayesian regularization methods have less difficulty in reaching convergence across chains (Jacobucci and Grimm 2017). With the recent advent of additional sparsity inducing priors, along with new forms of software such as Stan (Carpenter et al. 2016), for some models it may be more appropriate to use these Bayesian regularization methods over their frequentist counterparts. In the realm of Bayesian regularization for structural equation models, although some research exists (Feng, Wu, and Song 2017), much more is warranted.

Future research with **regsem** should focus on a number of avenues. One is comparing the different forms of regularization, delineating which method may be best in which setting. Additionally, as structural equation models become larger, with the advent of much larger datasets, computational speed will become a principal concern. Although 40 penalties in the models tested above can be run in a matter of seconds on a standard laptop, larger models can take much longer. To handle this, future implementation with **regsem** will test the inclusion of different types of optimization, specifically testing whether computation of the Hessian (or approximate Hessian) can reduce the number of optimization steps.

6.1. Conclusion

This paper provided a brief overview on the use of the **regsem** package as an implementation of regularized structural equation model. Because structural equation modeling encompasses a wide array of latent variable models, the **regsem** package was created as a general package for including different forms of regularization into a host of latent variable models. RegSEM, and thus the **regsem** package, has been evaluated in a wide array of SEM models, including confirmatory factor analysis (Jacobucci, Grimm, and McArdle 2016), latent change score models (Jacobucci and Grimm 2017), mediation models (Serang et al. 2017). Future updates to **regsem** will focus on decreasing the computational time of large latent variable models in order to provide an avenue of testing for researchers collecting larger and larger datasets. RegSEM is a method that operates at all ends of the data size spectrum: allowing for a reduction in complexity when the sample size is small, along with dimension reduction in the presence of large data (both N and P).

References

Brandmaier, Andreas M., Timo von Oertzen, John J. McArdle, and Ulman Lindenberger. 2013. "Structural Equation Model Trees." *Psychological Methods* 18 (1): 71–86.

Browne, Michael W. 2001. "An Overview of Analytic Rotation in Exploratory Factor

Analysis." *Multivariate Behavioral Research* 36 (1). Taylor & Francis: 111–50.

Carpenter, Bob, Andrew Gelman, Matt Hoffman, Daniel Lee, Ben Goodrich, Michael Betancourt, Michael A Brubaker, Jiqiang Guo, Peter Li, and Allen Riddell. 2016. "Stan: A Probabilistic Programming Language." *Journal of Statistical Software* 20.

Dahl, David B. 2009. "Xtable: Export Tables to Latex or Html." *R Package Version*, 1–5.

Donoho, David L. 1995. "De-Noising by Soft-Thresholding." *IEEE Transactions on Information Theory* 41 (3). IEEE: 613–27.

Epskamp, Sacha, Mijke Rhemtulla, and Denny Borsboom. 2016. "Generalized Network Psychometrics: Combining Network and Latent Variable Models." *ArXiv Preprint ArXiv:1605.09288.*

Fan, Jianqing, and Runze Li. 2001. "Variable Selection via Nonconcave Penalized Likelihood and Its Oracle Properties." *Journal of the American Statistical Association* 96 (456). Taylor & Francis: 1348–60.

Feng, Xiang-Nan, Hao-Tian Wu, and Xin-Yuan Song. 2017. "Bayesian Regularized Multivariate Generalized Latent Variable Models." *Structural Equation Modeling: A Multidisciplinary Journal.* Taylor & Francis, 1–18.

Hastie, Trevor, Robert Tibshirani, and Martin Wainwright. 2015. *Statistical Learning with Sparsity: The Lasso and Generalizations.* CRC Press.

Hoerl, Arthur E, and Robert W Kennard. 1970. "Ridge Regression: Biased Estimation for Nonorthogonal Problems." *Technometrics* 12 (1). Taylor & Francis Group: 55–67.

Holzinger, Karl John, and Frances Swineford. 1939. "A Study in Factor Analysis: The Stability of a Bi-Factor Solution." *Supplementary Educational Monographs.*

Jacobucci, Ross. 2017. "Package 'Regsem'." *Https://Cran.r-Project.org/Web/Packages/Regsem/Index.h*

Jacobucci, Ross, Kevin J Grimm, and John J McArdle. 2016. "Regularized Structural Equation Modeling." *Structural Equation Modeling: A Multidisciplinary Journal* 23 (4). Taylor & Francis: 555–66.

Kline, Rex B. 2015. *Principles and Practice of Structural Equation Modeling.* Guilford publications.

Lam, Clifford, and Jianqing Fan. 2009. "Sparsistency and Rates of Convergence in Large Covariance Matrix Estimation." *Annals of Statistics* 37 (6B). NIH Public Access: 4254.

Marsh, Herbert W, and Kit-Tai Hau. 1996. "Assessing Goodness of Fit: Is Parsimony Always Desirable?" *The Journal of Experimental Education* 64 (4). Taylor & Francis: 364–90.

McArdle, John J. 2005. "The Development of the Ram Rules for Latent Variable Structural Equation Modeling." *Contemporary Psychometrics: A Festschrift for Roderick P. McDonald.* Erlbaum. Mahwah, NJ, 225–73.

McArdle, John J, and Fumiaki Hamagami. 2001. "Latent Difference Score Structural Models for Linear Dynamic Analyses with Incomplete Longitudinal Data." American Psychological Association.

McArdle, John J., and Roderick P. McDonald. 1984. "Some Algebraic Properties of

the Reticular Action Model for Moment Structures." *British Journal of Mathematical and Statistical Psychology* 37 (2). Wiley-Blackwell: 234–51.

Meinshausen, Nicolai. 2007. "Relaxed Lasso." *Computational Statistics & Data Analysis* 52 (1). Elsevier: 374–93.

Meredith, William, and John Tisak. 1990. "Latent Curve Analysis." *Psychometrika* 55 (1). Springer: 107–22.

R Core Team. 2017. *R: A Language and Environment for Statistical Computing.* Vienna, Austria: R Foundation for Statistical Computing.

Raykov, Tenko, and George A Marcoulides. 1999. "On Desirability of Parsimony in Structural Equation Model Selection." *Structural Equation Modeling: A Multidisciplinary Journal* 6 (3). Taylor & Francis: 292–300.

Rosseel, Yves. 2012. "Lavaan: An R Package for Structural Equation Modeling." *Journal of Statistical Software* 48 (2): 1–36.

Schwarz, Gideon. 1978. "Estimating the Dimension of a Model." *The Annals of Statistics* 6 (2). Institute of Mathematical Statistics: 461–64.

Serang, Sarfaraz, Ross Jacobucci, Kim Brimhall, and Kevin J Grimm. 2017. "Exploratory Mediation Analysis via Regularization." *Structural Equation Modeling: A Multidisciplinary Journal.* Taylor & Francis.

Steiger, James H, and John C Lind. 1980. "Statistically Based Tests for the Number of Common Factors." In *Annual Meeting of the Psychometric Society, Iowa City, Ia.* Vol. 758.

Sun, Jianan, Yunxiao Chen, Jingchen Liu, Zhiliang Ying, and Tao Xin. 2016. "Latent Variable Selection for Multidimensional Item Response Theory Models via L_ {1} Regularization." *Psychometrika* 81 (4). Springer: 921–39.

Thurstone, L. L. 1935. *The Vectors of Mind.* Chicago, IL: University of Chicago Press.

Tibshirani, Robert. 1996. "Regression Shrinkage and Selection via the Lasso." *Journal of the Royal Statistical Society. Series B (Methodological)* 58 (1). Wiley for the Royal Statistical Society: 267–88.

Zhang, Cun-Hui. 2010. "Nearly Unbiased Variable Selection Under Minimax Concave Penalty." *The Annals of Statistics* 38 (2). Institute of Mathematical Statistics: 894–942.

Zou, Hui. 2006. "The Adaptive Lasso and Its Oracle Properties." *Journal of the American Statistical Association* 101 (476). Taylor & Francis: 1418–29.

Zou, Hui, and Trevor Hastie. 2005. "Regularization and Variable Selection via the Elastic Net." *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 67 (2). Wiley Online Library: 301–20.

Zou, Hui, Trevor Hastie, and Robert Tibshirani. 2006. "Sparse Principal Component Analysis." *Journal of Computational and Graphical Statistics* 15 (2). Taylor & Francis: 265–86.

———. 2007. "On the Degrees of Freedom of the Lasso." *The Annals of Statistics* 35 (5). Institute of Mathematical Statistics: 2173–92. doi:10.1214/009053607000000127.

Affiliation:

Ross Jacobucci
University of Notre Dame
118 Haggar Hall, Notre Dame, IN 46556
E-mail: rcjacobuc@gmail.com
URL: rjacobucci.com