



A Faster Procedure for Estimating SEMs Applying Minimum Distance Estimators With a Fixed Weight Matrix

David Kreiberg & Xingwu Zhou

To cite this article: David Kreiberg & Xingwu Zhou (2022): A Faster Procedure for Estimating SEMs Applying Minimum Distance Estimators With a Fixed Weight Matrix, Structural Equation Modeling: A Multidisciplinary Journal, DOI: [10.1080/10705511.2022.2076093](https://doi.org/10.1080/10705511.2022.2076093)

To link to this article: <https://doi.org/10.1080/10705511.2022.2076093>



© 2022 The Author(s). Published with license by Taylor & Francis Group, LLC



Published online: 10 Oct 2022.



Submit your article to this journal [↗](#)



Article views: 134



View related articles [↗](#)



View Crossmark data [↗](#)

A Faster Procedure for Estimating SEMs Applying Minimum Distance Estimators With a Fixed Weight Matrix

David Kreiberg^a and Xingwu Zhou^b

^aBI Norwegian Business School; ^bUppsala University

ABSTRACT

This study presents a separable nonlinear least squares (SNLLS) implementation of the minimum distance (MD) estimator employing a fixed-weight matrix for estimating structural equation models (SEMs). In contrast to the standard implementation of the MD estimator, in which the complete set of parameters is estimated using nonlinear optimization, the SNLLS implementation allows a subset of parameters to be estimated using (linear) least squares (LS). The SNLLS implementation possesses a number of benefits, such as faster convergence, better performance in ill-conditioned estimation problems, and fewer required starting values. The present work demonstrates that SNLLS, when applied to SEM estimation problems, significantly reduces the estimation time. Reduced estimation time makes SNLLS particularly useful in applications involving some form of resampling, such as simulation and bootstrapping.

KEYWORDS

Minimum distance estimation; numerical efficiency; quadratic form fit function; structural equation models

1. Introduction

This study addresses the application of separable nonlinear least squares (SNLLS) when performing covariance structure analysis (CSA). SNLLS was first introduced by Golub and Pereyra (1973), who showed that for a certain type of nonlinear estimation problems, a subset of parameters can be estimated using numerically efficient least squares (LS). As will be discussed below, several studies have shown that parameter separation offers a number of numerical benefits, such as faster convergence, better performance when the estimation problem is ill-conditioned (i.e., problems in which the ratio between the largest and the smallest singular value of the covariance matrix is large), and fewer required starting values.



SNLLS is typically applied to problems involving some form of nonlinear regression analysis, but not exclusively so. A recent study by Kreiberg et al. (2021) suggested an SNLLS implementation of the minimum distance (MD) estimator for estimating confirmatory factor analysis (CFA) models. The motivation for the current study is to generalize the results in Kreiberg et al. (2021) by outlining an SNLLS implementation for estimating structural equation models (SEMs). This is important for several reasons. First, it makes SNLLS applicable to a wider range of models. Second, at this stage, little is known about the potential benefits of applying SNLLS in the context of CSA. The outlined SNLLS implementation may pave the way for future research on

how to improve the numerical performance of CSA based estimators.

To make the idea of SNLLS clearer, consider the familiar MD quadratic form objective function

$$F(\boldsymbol{\vartheta}) = (\mathbf{s}_x - \boldsymbol{\sigma}_x(\boldsymbol{\vartheta}))^T \mathbf{V} (\mathbf{s}_x - \boldsymbol{\sigma}_x(\boldsymbol{\vartheta})), \quad (1)$$

where \mathbf{s}_x and $\boldsymbol{\sigma}_x(\boldsymbol{\vartheta})$ are covariance vectors derived from the sample and the model, respectively, $\boldsymbol{\vartheta}$ is the parameter vector and \mathbf{V} is a weighting matrix chosen by the user. We consider the case in which \mathbf{V} is a fixed matrix (i.e., when \mathbf{V} is not a function of $\boldsymbol{\vartheta}$). Such cases include well-known estimators such as unweighted least squares (ULS), generalized least squares (GLS), and weighted least squares (WLS). The standard implementation of Equation (1) is a one-step estimation procedure, here referred to as nonlinear least squares (NLLS), that involves the use of nonlinear optimization techniques. Estimation is performed by searching the parameter space for the value of $\boldsymbol{\vartheta}$ that minimizes Equation (1). In contrast, the SNLLS implementation of Equation (1) is a two-step estimation procedure that works by splitting $\boldsymbol{\vartheta}$ into two subsets. In the first step, one subset of parameters is estimated using nonlinear optimization. In the second step, based on the estimates obtained in the first step, the remaining subset of parameters is estimated using LS. As demonstrated in Kreiberg et al. (2021), SNLLS provides parameter estimates and a minimum objective function value identical to those obtained using NLLS. It obviously follows that the asymptotic properties of the estimator are

CONTACT Xingwu Zhou  xingwu.zhou@medsci.uu.se  Department of Medical Sciences, Clinical Physiology, Uppsala University, Akademiska sjukhuset, ingång 40, 3 tr, Uppsala, 751 85 Sweden; Department of Medical Sciences, Respiratory, Allergy and Sleep Research, Uppsala University, Uppsala, 751 05 Sweden; Department of Statistics, Uppsala University, Uppsala, 751 05 Sweden.

© 2022 The Author(s). Published with license by Taylor & Francis Group, LLC

This is an Open Access article distributed under the terms of the Creative Commons Attribution-NonCommercial-NoDerivatives License (<http://creativecommons.org/licenses/by-nc-nd/4.0/>), which permits non-commercial re-use, distribution, and reproduction in any medium, provided the original work is properly cited, and is not altered, transformed, or built upon in any way.

maintained. The presentation below presents a general framework for how to accomplish parameter separation in the case of SEMs.

Over the years, SNLLS has become popular in applied research across a wide range of scientific disciplines. Golub and Pereyra (2003) compiled a list of real-world examples of SNLLS applications. Mullen (2008) subsequently provided a comprehensive overview of SNLLS for a number of applications in physics and chemistry. SNLLS has also proved useful in systems and control applications. For instance, Söderström et al. (2009), Söderström & Mossberg (2011), and Kreiberg et al. (2016) applied CSA to handle the errors-in-variables (EIV) estimation problem. The work in these studies showed how to implement the MD estimator using SNLLS.

Several studies have documented that the SNLLS implementation of nonlinear estimators offers a number of benefits. For instance, Sjöberg and Viberg (1997) evaluated the numerical performance of SNLLS when applied to neural-network minimization problems. Their main conclusions were that SNLLS provides faster convergence and performs better in cases in which the estimation problem is ill-conditioned. A recent study by Dattner et al. (2020) investigated the performance of SNLLS when applied to estimation problems involving ordinary differential equations (ODEs). Their simulations showed that SNLLS provides faster convergence as well as parameter estimates of similar or higher accuracy than what is achieved by traditional nonlinear procedures.

The remainder of this article is organized as follows. Section 2 establishes the notation used throughout the article. In this section, we provide a brief overview of the SEM framework and the associated MD estimator. Section 3 outlines how to modify the MD objective function to accommodate the SNLLS implementation of the estimator when applied to SEMs. Section 4 compares the numerical efficiency of SNLLS and NLLS when applied to real-world estimation problems. Finally, Section 5 presents some concluding remarks.

2. Background

2.1. Notation

Before presenting the SEM framework, it will be useful to introduce the following notation. Let \mathbf{x} be a $p \times 1$ zero-mean random vector, and let Σ_x be the associated $p \times p$ covariance matrix given by

$$\Sigma_x = E[\mathbf{x}\mathbf{x}^T], \quad (2)$$

where E is the expectation operator and the superscript T is the transpose of a vector or a matrix. The number of non-redundant elements in Σ_x is $h = 2^{-1}p(p+1)$, given that no restrictions other than symmetry are placed on the elements of Σ_x . A covariance vector containing the nonredundant elements (i.e., the lower half of Σ_x including the diagonal) is

$$\boldsymbol{\sigma}_x = \text{vech}(\Sigma_x). \quad (3)$$

In this expression, vech is the operation of vectorizing the non-redundant elements of Σ_x . Alternatively, $\boldsymbol{\sigma}_x$ is obtained by

$$\boldsymbol{\sigma}_x = \mathbf{K}_x^T \text{vec}(\Sigma_x). \quad (4)$$

Here, vec is the operation of vectorizing the elements of a matrix by stacking its columns, and \mathbf{K}_x is a $p^2 \times h$ matrix obtained from

$$\mathbf{K}_x = \mathbf{L}_x (\mathbf{L}_x^T \mathbf{L}_x)^{-1}, \quad (5)$$

where \mathbf{L}_x is a $p^2 \times h$ selection matrix containing only ones and zeros. This matrix has the additional usage

$$\text{vec}(\Sigma_x) = \mathbf{L}_x \boldsymbol{\sigma}_x. \quad (6)$$

In the case of symmetry, \mathbf{L}_x is referred to as the duplication matrix in the literature (see Magnus & Neudecker, 1999). The matrices \mathbf{L}_x and \mathbf{K}_x can be formed to handle covariance matrices with additional structure beyond symmetry. For instance, in the case that Σ_x is a diagonal, \mathbf{K}_x is constructed so that $\boldsymbol{\sigma}_x$ contains only the elements on the diagonal of Σ_x . Appendix A outlines a general framework for how to obtain \mathbf{L}_x and \mathbf{K}_x for various structures characterizing Σ_x .

We now expand the previous notation. Let \mathbf{x}_1 and \mathbf{x}_2 be $p_1 \times 1$ and $p_2 \times 1$ zero-mean random vectors, respectively. A $p = p_1 + p_2$ dimensional column vector is given by

$$\mathbf{x} = (\mathbf{x}_1^T \ \mathbf{x}_2^T)^T. \quad (7)$$

The associated $p \times p$ covariance matrix is

$$\Sigma_x = \begin{pmatrix} \Sigma_{x_1} & \Sigma_{x_2, x_1}^T \\ \Sigma_{x_2, x_1} & \Sigma_{x_2} \end{pmatrix}, \quad (8)$$

where

$$\Sigma_{x_1} = E[\mathbf{x}_1 \mathbf{x}_1^T], \quad \Sigma_{x_2} = E[\mathbf{x}_2 \mathbf{x}_2^T], \quad \Sigma_{x_2, x_1} = E[\mathbf{x}_2 \mathbf{x}_1^T]. \quad (9)$$

As before, the vector consisting of the nonredundant elements of Σ_x is given by $\boldsymbol{\sigma}_x$. However, for later, it will be more convenient to work with the vector

$$\tilde{\boldsymbol{\sigma}}_x = (\boldsymbol{\sigma}_{x_1}^T \ \boldsymbol{\sigma}_{x_2}^T \ \boldsymbol{\sigma}_{x_2, x_1}^T)^T, \quad (10)$$

where

$$\boldsymbol{\sigma}_{x_1} = \mathbf{K}_{x_1}^T \text{vec}(\Sigma_{x_1}), \quad \boldsymbol{\sigma}_{x_2} = \mathbf{K}_{x_2}^T \text{vec}(\Sigma_{x_2}), \quad \boldsymbol{\sigma}_{x_2, x_1} = \text{vec}(\Sigma_{x_2, x_1}). \quad (11)$$

Note that $\tilde{\boldsymbol{\sigma}}_x$ contains the same elements as $\boldsymbol{\sigma}_x$, but in a different order. The last equation in Equation (11) follows from the fact that there is no redundancy in Σ_{x_2, x_1} . Appendix A shows how to derive a matrix $\tilde{\mathbf{L}}_x$. Then, by using Equations (4) and (5), we obtain the covariance vector

$$\tilde{\boldsymbol{\sigma}}_x = \tilde{\mathbf{K}}_x^T \text{vec}(\Sigma_x). \quad (12)$$

2.2. The SEM Framework

With the basic notation in place, we are ready to introduce the SEM framework, which consists of the following three equations (excluding constant terms)

$$\boldsymbol{\eta} = \mathbf{B}\boldsymbol{\eta} + \boldsymbol{\Gamma}\boldsymbol{\xi} + \boldsymbol{\delta}, \quad (13)$$

$$\mathbf{x}_1 = \mathbf{A}_1\boldsymbol{\eta} + \boldsymbol{\epsilon}_1, \quad (14)$$

$$\mathbf{x}_2 = \mathbf{A}_2\boldsymbol{\xi} + \boldsymbol{\epsilon}_2. \quad (15)$$

The first equation is the structural equation, which specifies the causal relationships among the latent variables. In this equation, $\boldsymbol{\eta}$ and $\boldsymbol{\xi}$ are respectively $p_\eta \times 1$ and $p_\xi \times 1$ random vectors, $\boldsymbol{\delta}$ is a $p_\eta \times 1$ random noise vector, and \mathbf{B} and $\boldsymbol{\Gamma}$ are respectively $p_\eta \times p_\eta$ and $p_\eta \times p_\xi$ parameter matrices relating the latent random vectors. The last two equations are measurement equations. In these equations, \mathbf{x}_1 and \mathbf{x}_2 are respectively $p_1 \times 1$ and $p_2 \times 1$ observed random vectors, $\boldsymbol{\epsilon}_1$ and $\boldsymbol{\epsilon}_2$ are noise vectors of similar dimensions, and \mathbf{A}_1 and \mathbf{A}_2 are respectively $p_1 \times p_\eta$ and $p_2 \times p_\xi$ parameter matrices relating the observed and the latent random vectors. All random vectors are zero-mean.

It is assumed that $\mathbf{I} - \mathbf{B}$, where \mathbf{I} is the identity matrix, is nonsingular such that $\boldsymbol{\eta}$ is uniquely determined by $\boldsymbol{\xi}$ and $\boldsymbol{\delta}$. It is further assumed that $\boldsymbol{\delta}$ and $\boldsymbol{\xi}$ are mutually uncorrelated, and that $\boldsymbol{\epsilon}_1$ and $\boldsymbol{\epsilon}_2$ are mutually uncorrelated with $\boldsymbol{\eta}$ and $\boldsymbol{\xi}$, respectively. The noise vectors $\boldsymbol{\epsilon}_1$ and $\boldsymbol{\epsilon}_2$ are allowed to correlate.

The specification additionally includes the following covariance matrices

$$\boldsymbol{\Sigma}_\xi = E[\boldsymbol{\xi}\boldsymbol{\xi}^T], \quad \boldsymbol{\Sigma}_\delta = E[\boldsymbol{\delta}\boldsymbol{\delta}^T], \quad \boldsymbol{\Sigma}_{\epsilon_1} = E[\boldsymbol{\epsilon}_1\boldsymbol{\epsilon}_1^T], \quad (16)$$

$$\boldsymbol{\Sigma}_{\epsilon_2} = E[\boldsymbol{\epsilon}_2\boldsymbol{\epsilon}_2^T], \quad \boldsymbol{\Sigma}_{\epsilon_2, \epsilon_1} = E[\boldsymbol{\epsilon}_2\boldsymbol{\epsilon}_1^T].$$

The nonredundant elements of $\boldsymbol{\Sigma}_\xi$, $\boldsymbol{\Sigma}_\delta$, $\boldsymbol{\Sigma}_{\epsilon_1}$, $\boldsymbol{\Sigma}_{\epsilon_2}$ and $\boldsymbol{\Sigma}_{\epsilon_2, \epsilon_1}$ are given by the covariance vectors

$$\boldsymbol{\sigma}_\xi = \mathbf{K}_\xi^T \text{vec}(\boldsymbol{\Sigma}_\xi), \quad \boldsymbol{\sigma}_\delta = \mathbf{K}_\delta^T \text{vec}(\boldsymbol{\Sigma}_\delta), \quad \boldsymbol{\sigma}_{\epsilon_1} = \mathbf{K}_{\epsilon_1}^T \text{vec}(\boldsymbol{\Sigma}_{\epsilon_1}), \quad (17)$$

$$\boldsymbol{\sigma}_{\epsilon_2} = \mathbf{K}_{\epsilon_2}^T \text{vec}(\boldsymbol{\Sigma}_{\epsilon_2}), \quad \boldsymbol{\sigma}_{\epsilon_2, \epsilon_1} = \text{vec}(\boldsymbol{\Sigma}_{\epsilon_2, \epsilon_1}).$$

Let $\boldsymbol{\vartheta}$ be a parameter vector containing the free elements in \mathbf{B} , $\boldsymbol{\Gamma}$, \mathbf{A}_1 , \mathbf{A}_2 , $\boldsymbol{\Sigma}_\xi$, $\boldsymbol{\Sigma}_\delta$, $\boldsymbol{\Sigma}_{\epsilon_1}$, $\boldsymbol{\Sigma}_{\epsilon_2}$ and $\boldsymbol{\Sigma}_{\epsilon_2, \epsilon_1}$, and let $\mathbf{H} = (\mathbf{I} - \mathbf{B})^{-1}$. The covariance matrix implied by Equations (13)–(15) is

$$\boldsymbol{\Sigma}_x(\boldsymbol{\vartheta}) = \begin{pmatrix} \mathbf{A}_1\mathbf{H}(\boldsymbol{\Gamma}\boldsymbol{\Sigma}_\xi\boldsymbol{\Gamma}^T + \boldsymbol{\Sigma}_\delta)\mathbf{H}^T\mathbf{A}_1^T + \boldsymbol{\Sigma}_{\epsilon_1} & \mathbf{A}_1\mathbf{H}\boldsymbol{\Gamma}\boldsymbol{\Sigma}_\xi\mathbf{A}_2^T + \boldsymbol{\Sigma}_{\epsilon_2, \epsilon_1}^T \\ \mathbf{A}_2\boldsymbol{\Sigma}_\xi\boldsymbol{\Gamma}^T\mathbf{H}^T\mathbf{A}_1^T + \boldsymbol{\Sigma}_{\epsilon_2, \epsilon_1} & \mathbf{A}_2\boldsymbol{\Sigma}_\xi\mathbf{A}_2^T + \boldsymbol{\Sigma}_{\epsilon_2} \end{pmatrix}. \quad (18)$$

2.3. The MD Estimator

Suppose that a sample of data points \mathbf{x}_i (for $i = 1, \dots, N$) is available. An estimate of $\boldsymbol{\Sigma}_x$ is then computed using

$$\mathbf{S}_x = \frac{1}{N} \sum_{i=1}^N \mathbf{x}_i \mathbf{x}_i^T. \quad (19)$$

Given \mathbf{S}_x , the aim is to estimate the true parameter vector $\boldsymbol{\vartheta}_0$. An estimate of $\boldsymbol{\vartheta}_0$ is obtained by

$$\hat{\boldsymbol{\vartheta}} = \arg \min_{\boldsymbol{\vartheta}} F(\boldsymbol{\vartheta}), \quad (20)$$

where $F(\boldsymbol{\vartheta})$ is a scalar function that expresses the distance between the observed and the model-implied covariance structure. Below, we focus on the MD objective function given by

$$F(\boldsymbol{\vartheta}) = (\mathbf{s}_x - \boldsymbol{\sigma}_x(\boldsymbol{\vartheta}))^T \mathbf{V} (\mathbf{s}_x - \boldsymbol{\sigma}_x(\boldsymbol{\vartheta})). \quad (21)$$

In this expression, \mathbf{s}_x and $\boldsymbol{\sigma}_x(\boldsymbol{\vartheta})$ are vectors containing the nonredundant elements of \mathbf{S}_x and $\boldsymbol{\Sigma}_x(\boldsymbol{\vartheta})$, respectively. That is,

$$\mathbf{s}_x = \mathbf{K}_x^T \text{vec}(\mathbf{S}_x), \quad \boldsymbol{\sigma}_x(\boldsymbol{\vartheta}) = \mathbf{K}_x^T \text{vec}(\boldsymbol{\Sigma}_x(\boldsymbol{\vartheta})). \quad (22)$$

Moreover, the matrix \mathbf{V} is a positive definite weighting matrix. Under suitable conditions, and for the right choice of \mathbf{V} , the MD estimator is consistent and asymptotically normal. Note that consistency does not depend on \mathbf{V} as long as \mathbf{V} converges in probability to a symmetric positive definite matrix.

Using a proper algorithm, Equation (21) is minimized by numerically searching the parameter space until some convergence criterion is satisfied. For the estimation problem to be feasible, it is a necessary condition that the number of elements in $\mathbf{s}_x - \boldsymbol{\sigma}_x(\boldsymbol{\vartheta})$ is at least as large as the number of free parameters in $\boldsymbol{\vartheta}$.

3. Modifying the MD Quadratic Form Objective Function

Next, we outline how to modify the objective function in Equation (21) to accommodate the SNLLS implementation. To do so, we need some additional notation. Let $\boldsymbol{\vartheta}_{\beta, \gamma, \lambda}$ be a $t_{\boldsymbol{\vartheta}_{\beta, \gamma, \lambda}} \times 1$ vector containing the free elements in \mathbf{B} , $\boldsymbol{\Gamma}$, \mathbf{A}_1 , and \mathbf{A}_2 , and let $\boldsymbol{\sigma}_{\xi, \delta, \epsilon}$ be a $t_{\boldsymbol{\sigma}_{\xi, \delta, \epsilon}} \times 1$ vector containing the free elements in $\boldsymbol{\Sigma}_\xi$, $\boldsymbol{\Sigma}_\delta$, $\boldsymbol{\Sigma}_{\epsilon_1}$, $\boldsymbol{\Sigma}_{\epsilon_2}$, and $\boldsymbol{\Sigma}_{\epsilon_2, \epsilon_1}$. The vector $\boldsymbol{\sigma}_{\xi, \delta, \epsilon}$ is formed by

$$\boldsymbol{\sigma}_{\xi, \delta, \epsilon} = \left(\boldsymbol{\sigma}_\xi^T \quad \boldsymbol{\sigma}_\delta^T \quad \boldsymbol{\sigma}_{\epsilon_1}^T \quad \boldsymbol{\sigma}_{\epsilon_2}^T \quad \boldsymbol{\sigma}_{\epsilon_2, \epsilon_1}^T \right)^T. \quad (23)$$

The complete parameter vector now becomes

$$\boldsymbol{\vartheta} = \left(\boldsymbol{\vartheta}_{\beta, \gamma, \lambda}^T \quad \boldsymbol{\sigma}_{\xi, \delta, \epsilon}^T \right)^T. \quad (24)$$

The key to applying SNLLS is the separation of parameters, which involves expressing the covariance vector in Equation (10) using

$$\tilde{\boldsymbol{\sigma}}_x = \mathbf{G}(\boldsymbol{\vartheta}_{\beta, \gamma, \lambda}) \boldsymbol{\sigma}_{\xi, \delta, \epsilon}. \quad (25)$$

In this expression, $\mathbf{G}(\boldsymbol{\vartheta}_{\beta, \gamma, \lambda})$ is a tall matrix valued function (i.e., a matrix consisting of more rows than columns) assumed to have full column rank. In Appendix B, it is shown that $\mathbf{G}(\boldsymbol{\vartheta}_{\beta, \gamma, \lambda})$ takes the general form

$$\mathbf{G}(\boldsymbol{\vartheta}_{\beta, \gamma, \lambda}) = \begin{pmatrix} \mathbf{K}_{x_1}^T (\mathbf{A}_1 \mathbf{H} \boldsymbol{\Gamma} \otimes \mathbf{A}_1 \mathbf{H} \boldsymbol{\Gamma}) \mathbf{L}_\xi & \mathbf{K}_{x_1}^T (\mathbf{A}_1 \mathbf{H} \otimes \mathbf{A}_1 \mathbf{H}) \mathbf{L}_\delta & \mathbf{K}_{x_1}^T \mathbf{L}_{\epsilon_1} & \mathbf{0} & \mathbf{0} \\ \mathbf{K}_{x_2}^T (\mathbf{A}_2 \otimes \mathbf{A}_2) \mathbf{L}_\xi & \mathbf{0} & \mathbf{0} & \mathbf{K}_{x_2}^T \mathbf{L}_{\epsilon_2} & \mathbf{0} \\ (\mathbf{A}_1 \mathbf{H} \boldsymbol{\Gamma} \otimes \mathbf{A}_2) \mathbf{L}_\xi & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} \end{pmatrix}, \quad (26)$$

where \otimes is the Kronecker product, the $\mathbf{0}$ s are zero matrices of compatible sizes, and \mathbf{I} is the identity matrix. It is now possible to write the objective function using

$$F(\boldsymbol{\vartheta}_{\beta, \gamma, \lambda}, \boldsymbol{\sigma}_{\xi, \delta, \epsilon}) = (\tilde{\mathbf{s}}_x - \mathbf{G}(\boldsymbol{\vartheta}_{\beta, \gamma, \lambda}) \boldsymbol{\sigma}_{\xi, \delta, \epsilon})^T \tilde{\mathbf{V}} (\tilde{\mathbf{s}}_x - \mathbf{G}(\boldsymbol{\vartheta}_{\beta, \gamma, \lambda}) \boldsymbol{\sigma}_{\xi, \delta, \epsilon}), \quad (27)$$

where $\tilde{\mathbf{s}}_x$ and $\tilde{\mathbf{V}}$ correspond to \mathbf{s}_x and \mathbf{V} , respectively, but with their rows and columns rearranged according to the order in $\tilde{\boldsymbol{\sigma}}_x$. For some value of $\mathbf{G}(\boldsymbol{\vartheta}_{\beta,\gamma,\lambda})$, the solution to the problem of minimizing Equation (27) w.r.t. $\boldsymbol{\sigma}_{\xi,\delta,\epsilon}$ is a straightforward application of LS

$$\hat{\boldsymbol{\sigma}}_{\xi,\delta,\epsilon}(\boldsymbol{\vartheta}_{\beta,\gamma,\lambda}) = \left(\mathbf{G}^T(\boldsymbol{\vartheta}_{\beta,\gamma,\lambda}) \tilde{\mathbf{V}} \mathbf{G}(\boldsymbol{\vartheta}_{\beta,\gamma,\lambda}) \right)^{-1} \mathbf{G}^T(\boldsymbol{\vartheta}_{\beta,\gamma,\lambda}) \tilde{\mathbf{V}} \tilde{\mathbf{s}}_x. \quad (28)$$

Since $\hat{\boldsymbol{\sigma}}_{\xi,\delta,\epsilon}$ depends on $\boldsymbol{\vartheta}_{\beta,\gamma,\lambda}$, it is necessary to outline how to obtain an estimate $\hat{\boldsymbol{\vartheta}}_{\beta,\gamma,\lambda}$ without directly involving $\boldsymbol{\sigma}_{\xi,\delta,\epsilon}$. Theorem 2.1 in Golub and Pereyra (1973) provides the justification for replacing $\boldsymbol{\sigma}_{\xi,\delta,\epsilon}$ in Equation (27) with the right-hand side of Equation (28). Doing so, leads to the modified objective function

$$F(\boldsymbol{\vartheta}_{\beta,\gamma,\lambda}) = \tilde{\mathbf{s}}_x^T \tilde{\mathbf{V}} \tilde{\mathbf{s}}_x - \tilde{\mathbf{s}}_x^T \tilde{\mathbf{V}} \mathbf{G}(\boldsymbol{\vartheta}_{\beta,\gamma,\lambda}) \left(\mathbf{G}^T(\boldsymbol{\vartheta}_{\beta,\gamma,\lambda}) \tilde{\mathbf{V}} \mathbf{G}(\boldsymbol{\vartheta}_{\beta,\gamma,\lambda}) \right)^{-1} \mathbf{G}^T(\boldsymbol{\vartheta}_{\beta,\gamma,\lambda}) \tilde{\mathbf{V}} \tilde{\mathbf{s}}_x. \quad (29)$$

Apart from some slight notational differences, the derivation of Equation (29) is similar to the derivation in Kreiberg et al. (2021). From the preceding presentation, it follows that SNLLS is a two-step procedure. In the first step, $\hat{\boldsymbol{\vartheta}}_{\beta,\gamma,\lambda}$ is obtained by minimizing Equation (29) applying nonlinear optimization. In the second step, using $\hat{\boldsymbol{\vartheta}}_{\beta,\gamma,\lambda}$ from the first step, $\hat{\boldsymbol{\sigma}}_{\xi,\delta,\epsilon}$ is obtained by Equation (28).

The major benefit of the formulation in Equation (29) is that the minimization w.r.t. $\boldsymbol{\vartheta}_{\beta,\gamma,\lambda}$ represents a lower dimensional optimization problem. Thus, the computational load when minimizing $F(\boldsymbol{\vartheta}_{\beta,\gamma,\lambda})$ w.r.t. $\boldsymbol{\vartheta}_{\beta,\gamma,\lambda}$ is smaller, and in some cases by a considerable margin, than what is the case when minimizing Equation (21) w.r.t. $\boldsymbol{\vartheta}$. This is especially the case when the number of elements in $\boldsymbol{\sigma}_{\xi,\delta,\epsilon}$ is large compared with the number of elements in $\boldsymbol{\vartheta}_{\beta,\gamma,\lambda}$.

4. Illustrations

This section provides two examples that illustrate the difference in numerical efficiency between the two implementations, SNLLS and NLLS, of the MD estimator when applied to SEMs. Numerical performance is assessed by studying the convergence of the optimizer and the time it takes the optimizer to reach its minimum. Since timing depends on other processes running on the device performing the estimation, it is recommended to compute the average estimation time over multiple runs. Estimation and timing are performed using Matlab (2020, version R2020b). The two implementations are compared under the following conditions:

- **Algorithm:** The optimizer is a Quasi-Newton (QN) design applying the Broyden–Fletcher–Goldfarb–Shanno (BFGS) Hessian update mechanism (default in Matlab).
- **Gradient:** For simplicity, the gradient is computed using a finite difference approach. The computation is based on a centered design, which is supposed to provide greater accuracy at the expense of being more time-consuming.

- **Tolerances:** Tolerances are set to their default values (details are found in the Matlab documentation).
- **Starting values:** Starting values are taken from the open-source R (R Core Team, 2021) package lavaan (Rosseel, 2012). The starting values for the free elements are as follows:

- \mathbf{A}_1 and \mathbf{A}_2 are computed using the non-iterative *fabin* 3 estimator (see Häggglund, 1982).
- \mathbf{B} and $\mathbf{\Gamma}$ are set to zero.
- $\boldsymbol{\Sigma}_\xi$ and $\boldsymbol{\Sigma}_\delta$ are set to zero except for the diagonal elements, which are set to 0.05.
- $\boldsymbol{\Sigma}_{\epsilon_1}$ and $\boldsymbol{\Sigma}_{\epsilon_2}$ are set to zero except for the diagonal elements, which are set to half the observed variance. For the examples below, no starting values are required for the elements in $\boldsymbol{\Sigma}_{\epsilon_2,\epsilon_1}$.

Note that SNLLS only requires starting values for the parameter vector $\boldsymbol{\vartheta}_{\beta,\gamma,\lambda}$, whereas NLLS requires starting values for the complete parameter vector $\boldsymbol{\vartheta}$.

- **Estimator:** The GLS estimator is used throughout the examples. The GLS estimator uses a weight matrix of the form

$$\tilde{\mathbf{V}} = 2^{-1} \tilde{\mathbf{L}}_x^T (\mathbf{S}^{-1} \otimes \mathbf{S}^{-1}) \tilde{\mathbf{L}}_x. \quad (30)$$

- **Timing:** In each example, the model is re-estimated 1000 times using the same empirical covariance matrix as input.

To ensure that our programming is correct, we compared the estimation results to the results obtained using lavaan.

4.1. Example 1

The first example considers a model for the medical illness of depression. The data ($N = 323$) used in this example are taken from Geiser (2012) and consist of six indicators of depression. In the data, $X_{1,1}$ and $X_{1,2}$ are indicators of the first-order common factor *Depression State 1*, $X_{1,3}$ and $X_{1,4}$ are indicators of the first-order common factor *Depression State 2*, and $X_{1,5}$ and $X_{1,6}$ are indicators of the first-order common factor *Depression State 3*. The three factors themselves are indicators of the second-order common trait factor *Depression*. The model additionally contains an indicator-specific factor labeled *IS*. Indicators $X_{1,1}$, $X_{1,2}$, $X_{1,3}$, $X_{1,5}$, and the factor *Depression State 1* serve as marker variables. The path diagram illustrating the structure of the model is shown in Figure 1.

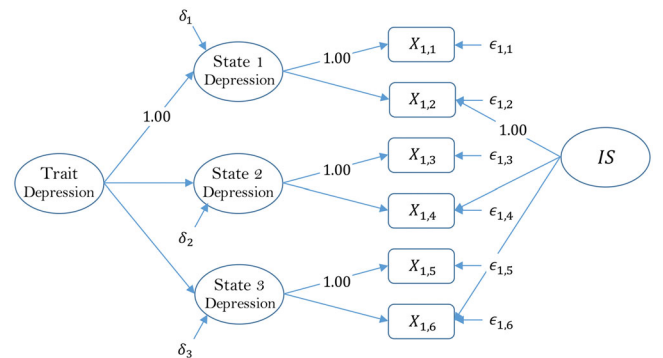


Figure 1. Geiser (2012).

Results of the estimation are presented in Table 1. As seen from the table, the number of iterations and function evaluations is $(It, Fe) = (23, 375)$ for SNLLS and $(It, Fe) = (145, 5439)$ for NLLS. As expected, the required computational load for minimizing $F(\vartheta_{\beta,\gamma,\lambda})$ w.r.t. $\vartheta_{\beta,\gamma,\lambda}$ is far less than the required load for minimizing $F(\vartheta)$ w.r.t. ϑ . Figure 2 shows the convergence profiles for the two implementations. From the figure, it is clear that the SNLLS objective function $F(\vartheta_{\beta,\gamma,\lambda})$ starts at a point much closer to its minimum of 0.0109 than what is seen for the NLLS objective function $F(\vartheta)$. In terms of estimation time, the mean time is 0.0334sec. for SNLLS and 0.1408 sec. for NLLS. Thus, SNLLS is faster by a factor of $0.1408/0.0334 = 4.2153$. The results in this example clearly suggest that the SNLLS implementation is numerically more efficient than the standard NLLS implementation.

4.2. Example 2

The second example considers a model for industrialization and political democracy. The model is taken from Bollen (1989), and has been used extensively in books, tutorials, etc. The data consist of 11 indicators of industrialization and political democracy for 75 countries ($N = 75$). In the data, $X_{1,1}, \dots, X_{1,4}$ are indicators of the common factor *Political Democracy* at time 1 (1960), $X_{1,5}, \dots, X_{1,8}$ are indicators of the common factor *Political Democracy* at time 2 (1965) and $X_{2,1}, \dots, X_{2,3}$ are indicators of the common factor *Industrialization* at time 1 (1960). Due to the repeated measurement design, the unique factors belonging to $X_{1,i}$ and $X_{1,i+4}$ for $i = 1, \dots, 4$ are set to correlate. Additionally, the unique factors belonging to $X_{1,i}$ and $X_{1,i+2}$ for $i = 2, 6$ are set

to correlate. Indicators $X_{1,1}, X_{1,5}$ and $X_{2,1}$ serve as marker variables. The path diagram of the model is shown in Figure 3.

Results of the estimation are presented in Table 2. The results in this example generally confirm the results from the previous example. In this case, the number of iterations and function evaluations are $(It, Fe) = (26, 759)$ for SNLLS and $(It, Fe) = (230, 14742)$ for NLLS. Figure 4 shows the convergence profiles for the two implementations. The patterns in the figure resemble those in Figure 2. Considering the estimation time, the mean time is 0.1257sec. for SNLLS and 0.8263 sec. for NLLS. In this case, SNLLS proves to be faster by a factor of $0.8263/0.1257 = 6.5736$.

5. Concluding Remarks

In this study, we have presented an SNLLS implementation of the MD objective function for estimating SEMs. The outlined framework includes all necessary expressions for applying SNLLS, and represents a generalization of

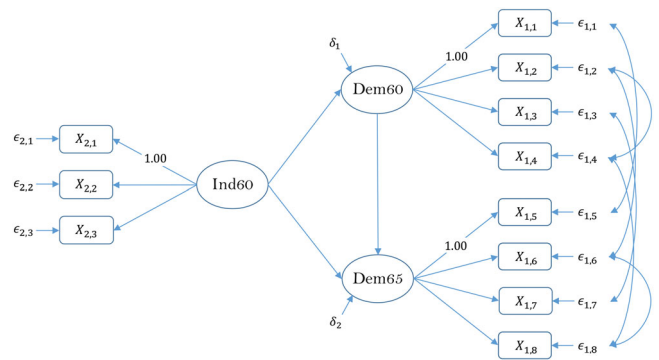


Figure 3. Bollen (1989).

Table 1. Timing results, Geiser (2012).

	SNLLS	NLLS
Mean est. time (in sec.)	0.0334	0.1408
Median est. time in (sec.)	0.0323	0.1393
Standard dev. est. time in (sec.)	0.0047	0.0067
Minimum objective func. value	0.0109	0.0109
Number of iterations (It)	23	145
Number func. evaluations (Fe)	375	5439
$t_{\sigma_{\beta,\gamma,\lambda}}/t_{\vartheta_{\beta,\gamma,\lambda}}$	1.57	1.57

Table 2. Timing results, Bollen (1989).

	SNLLS	NLLS
Mean est. time (in sec.)	0.1257	0.8263
Median est. time in (sec.)	0.1244	0.8246
Standard dev. est. time in (sec.)	0.0106	0.0182
Minimum objective func. value	0.4858	0.4858
Number of iterations (It)	26	230
Number func. evaluations (Fe)	759	14742
$t_{\sigma_{\beta,\gamma,\lambda}}/t_{\vartheta_{\beta,\gamma,\lambda}}$	1.82	1.82

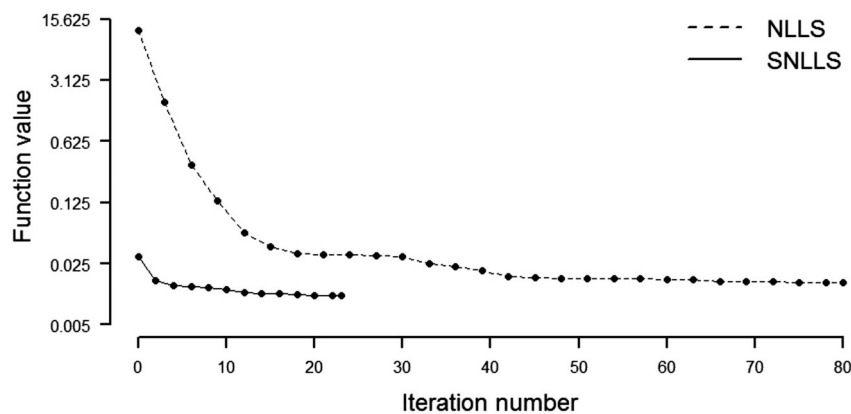


Figure 2. Convergence profile, Geiser (2012).

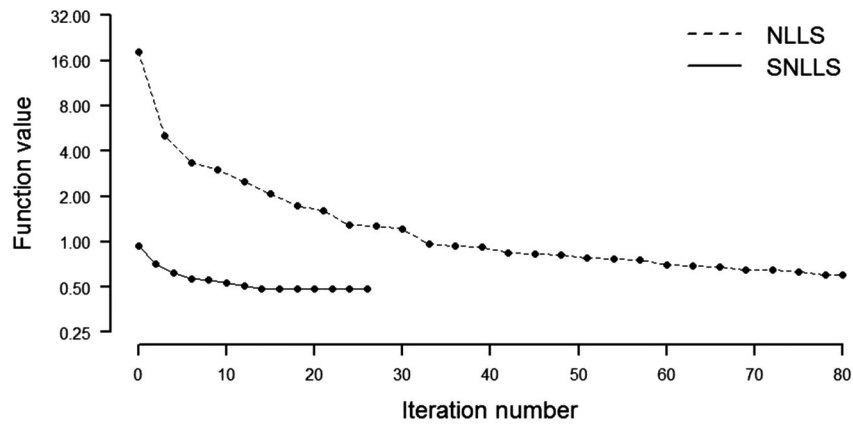


Figure 4. Convergence profile, Bollen (1989).

previously known results. Using examples from the SEM literature, we demonstrated that the computational load of applying SNLLS is considerably less than that of applying NLLS. Another benefit of SNLLS is that fewer starting values are required, which may mitigate potential problems due to the somewhat arbitrary choice of starting values for the covariance parameters.

The present work may have several interesting extensions. First, as shown by research, SNLLS may hold a potential for improving numerical performance in situations in which the estimation problem is ill-conditioned. Thus, an interesting case for future research would be to compare the numerical performance of SNLLS and NLLS under more challenging conditions in which the condition number of the observed covariance matrix is large. Second, the SNLLS implementation is not yet available for maximum likelihood (ML) estimation. Some initial work on this topic is underway. This work, combined with the previous point, may lead to an improved implementation of the ML estimator when applied to SEMs.

References

- Bollen, K. A. (1989). *Structural equations with latent variables*. Wiley.
- Dattner, I., Ship, H., & Voit, E. O. (2020). Separable nonlinear least-squares parameter estimation for complex dynamic systems. *Complexity*, 2020, 1–11. <https://doi.org/10.1155/2020/6403641>
- Geiser, C. (2012). *Data analysis with Mplus*. Guilford Press.
- Golub, G. H., & Pereyra, V. (1973). The differentiation of pseudo-inverses and nonlinear least squares problems whose variables separate. *SIAM Journal on Numerical Analysis*, 10, 413–432. <https://doi.org/10.1137/0710036>
- Golub, G. H., & Pereyra, V. (2003). Separable nonlinear least squares: The variable projection method and its applications. *Inverse Problems*, 19, R1–R26. <https://doi.org/10.1088/0266-5611/19/2/01>
- Häggglund, G. (1982). Factor analysis by instrumental variables methods. *Psychometrika*, 47, 209–222. <https://doi.org/10.1007/BF02296276>
- Kreiberg, D., Marcoulides, K., & Olsson, U. H. (2021). A faster procedure for estimating CFA models applying minimum distance estimators with a fixed weight matrix. *Structural Equation Modeling: A Multidisciplinary Journal*, 28, 725–739. <https://doi.org/10.1080/10705511.2020.1835484>
- Kreiberg, D., Söderström, T., & Yang-Wallentin, F. (2016). Errors-in-variables system identification using structural equation modeling. *Automatica*, 66, 218–230. <https://doi.org/10.1016/j.automatica.2015.12.007>

- Magnus, J. R., & Neudecker, H. (1999). *Matrix differential calculus with applications in statistics and econometrics*. John Wiley & Sons.
- MATLAB version R2020b. (2020). The MathWorks Inc.
- Mullen, K. M. (2008). *Separable nonlinear models: Theory, implementation and applications in physics and chemistry* [Ph.D. thesis]. Vrije Universiteit Amsterdam.
- R Core Team (2021). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing. <http://www.R-project.org/>
- Rosseel, Y. (2012). lavaan: An R package for structural equation modeling. *Journal of Statistical Software*, 48, 1–36. <https://doi.org/10.18637/jss.v048.i02>
- Sjöberg, J., & Viberg, M. (1997). Separable non-linear least-squares minimization-possible improvements for neural net fitting. In *Neural networks for signal processing VII. Proceedings of the 1997 IEEE signal processing society workshop* (pp. 345–354). IEEE.
- Söderström, T., & Mossberg, M. (2011). Accuracy analysis of a covariance matching approach for identifying errors-in-variables systems. *Automatica*, 47, 272–282. <https://doi.org/10.1016/j.automatica.2010.10.046>
- Söderström, T., Mossberg, M., & Hong, M. (2009). A covariance matching approach for identifying errors-in-variables systems. *Automatica*, 45, 2018–2031. <https://doi.org/10.1016/j.automatica.2009.05.010>

Appendices

A. Deriving L and K

Let x be an arbitrary $p \times 1$ random vector, and let $\Sigma_x = \{\sigma_{i,j}\}$ be the associated $p \times p$ covariance matrix. The purpose of the following presentation is to introduce an algebraic framework that facilitates eliminating the redundancy originating from the structure of Σ_x . To do so, let K_x be a matrix such that

$$\sigma_x = K_x^T \text{vec}(\Sigma_x), \quad (\text{A1})$$

where σ_x is a covariance vector containing the nonredundant elements of Σ_x and K_x is a matrix obtained by

$$K_x = L_x (L_x^T L_x)^{-1}. \quad (\text{A2})$$

In this expression, L_x is a selection matrix (i.e., a matrix composed of zeros and ones).

Below, we propose a rather general framework that applies to any structure characterizing Σ_x . Before presenting some examples on how to obtain L_x , it is necessary to introduce some additional notation. Let $E(u, v) = \{e_{i,j}(u, v)\}$ denote a $p \times p$ matrix (for $i, j = 1, \dots, p$) with elements

$$e_{i,j}(u, v) = \begin{cases} 1 & \text{if } \sigma_{i,j} = \sigma_{u,v} \\ 0 & \text{otherwise} \end{cases}. \quad (\text{A3})$$

Next, we demonstrate how to obtain \mathbf{L}_x for two standard cases and one case specialized for the SNLLS implementation.

Case 1: As a start, consider the case in which Σ_x is symmetric and no other restrictions are placed on its elements. The covariance vector containing the $2^{-1}p(p+1)$ nonredundant elements of Σ_x is

$$\boldsymbol{\sigma}_x = (\sigma_{1,1} \dots \sigma_{p,1} \sigma_{2,2} \dots \sigma_{p,2} \sigma_{3,3} \dots \dots \sigma_{p,p})^T. \quad (\text{A4})$$

Applying (A3), the matrix \mathbf{L}_x is formed by horizontally concatenating $2^{-1}p(p+1)$ vectors using

$$\mathbf{L}_x = \left(\text{vec}(\mathbf{E}(1,1)) \dots \text{vec}(\mathbf{E}(p,1)) \text{vec}(\mathbf{E}(2,2)) \dots \text{vec}(\mathbf{E}(p,2)) \right. \\ \left. \text{vec}(\mathbf{E}(3,3)) \dots \dots \text{vec}(\mathbf{E}(p,p)) \right). \quad (\text{A5})$$

Case 2: Now, consider the case in which Σ_x is diagonal. The covariance vector containing the p nonredundant elements of Σ_x is given by

$$\boldsymbol{\sigma}_x = (\sigma_{1,1} \sigma_{2,2} \dots \sigma_{p,p})^T. \quad (\text{A6})$$

The matrix \mathbf{L}_x is now formed by horizontally concatenating p vectors

$$\mathbf{L}_x = \left(\text{vec}(\mathbf{E}(1,1)) \text{vec}(\mathbf{E}(2,2)) \dots \text{vec}(\mathbf{E}(p,p)) \right). \quad (\text{A7})$$

Before introducing the third and final case, it is necessary to expand the notation. Let \mathbf{x}_1 and \mathbf{x}_2 be respectively $p_1 \times 1$ and $p_2 \times 1$ random vectors, and let \mathbf{x} be a $p = p_1 + p_2$ dimensional column vector obtained by stacking \mathbf{x}_1 and \mathbf{x}_2 in the following way

$$\mathbf{x} = (\mathbf{x}_1^T \ \mathbf{x}_2^T)^T. \quad (\text{A8})$$

The associated $p \times p$ covariance matrix is given by

$$\Sigma_x = \begin{pmatrix} \Sigma_{x_1} & \Sigma_{x_2, x_1}^T \\ (p_1 \times p_1) & (p_1 \times p_2) \\ \Sigma_{x_2, x_1} & \Sigma_{x_2} \\ (p_2 \times p_1) & (p_2 \times p_2) \end{pmatrix}. \quad (\text{A9})$$

Case 3: As in the first case, suppose that no other restrictions, apart from symmetry, are placed on the elements of Σ_x . Let the covariance vector containing the $2^{-1}p(p+1)$ nonredundant elements of Σ_x be given by

$$\tilde{\boldsymbol{\sigma}}_x = \left(\boldsymbol{\sigma}_{x_1}^T \ \boldsymbol{\sigma}_{x_2}^T \ \boldsymbol{\sigma}_{x_2, x_1}^T \right)^T, \quad (\text{A10})$$

where

$$\boldsymbol{\sigma}_{x_1} = (\sigma_{1,1} \dots \sigma_{p_1,1} \sigma_{2,2} \dots \sigma_{p_1,2} \sigma_{3,3} \dots \dots \sigma_{p_1,p_1})^T, \quad (\text{A11})$$

$$\boldsymbol{\sigma}_{x_2} = (\sigma_{p_1+1,p_1+1} \dots \sigma_{p,p_1+1} \sigma_{p_1+2,p_1+2} \dots \sigma_{p,p_1+2} \sigma_{p_1+3,p_1+3} \dots \dots \sigma_{p,p})^T, \quad (\text{A12})$$

$$\boldsymbol{\sigma}_{x_2, x_1} = (\sigma_{p_1+1,1} \dots \sigma_{p,1} \sigma_{p_1+1,2} \dots \sigma_{p,2} \sigma_{p_1+1,3} \dots \dots \sigma_{p,p_1})^T. \quad (\text{A13})$$

Construct a matrix $\tilde{\mathbf{L}}_x$ by horizontally concatenating three matrices

$$\tilde{\mathbf{L}}_x = \left((\tilde{\mathbf{L}}_x)_{1,1} \ (\tilde{\mathbf{L}}_x)_{1,2} \ (\tilde{\mathbf{L}}_x)_{1,3} \right), \quad (\text{A14})$$

where the submatrices are given by

$$(\tilde{\mathbf{L}}_x)_{1,1} = \left(\text{vec}(\mathbf{E}(1,1)) \dots \text{vec}(\mathbf{E}(p_1,1)) \text{vec}(\mathbf{E}(2,2)) \dots \text{vec}(\mathbf{E}(p_1,2)) \right. \\ \left. \text{vec}(\mathbf{E}(3,3)) \dots \dots \text{vec}(\mathbf{E}(p_1,p_1)) \right), \quad (\text{A15})$$

$$(\tilde{\mathbf{L}}_x)_{1,2} = \left(\text{vec}(\mathbf{E}(p_1+1,p_1+1)) \dots \text{vec}(\mathbf{E}(p,p_1+1)) \right. \\ \left. \text{vec}(\mathbf{E}(p_1+2,p_1+2)) \dots \text{vec}(\mathbf{E}(p,p_1+2)) \right. \\ \left. \text{vec}(\mathbf{E}(p_1+3,p_1+3)) \dots \dots \text{vec}(\mathbf{E}(p,p)) \right), \quad (\text{A16})$$

$$(\tilde{\mathbf{L}}_x)_{1,3} = \left(\text{vec}(\mathbf{E}(p_1+1,1)) \dots \text{vec}(\mathbf{E}(p,1)) \text{vec}(\mathbf{E}(p_1+1,2)) \dots \text{vec}(\mathbf{E}(p,2)) \right. \\ \left. \text{vec}(\mathbf{E}(p_1+1,3)) \dots \dots \text{vec}(\mathbf{E}(p,p_1)) \right). \quad (\text{A17})$$

The number of columns in (A15), (A16), and (A17) is $2^{-1}p_1(p_1+1)$, $2^{-1}p_2(p_2+1)$ and $p_2 \times p_1$, respectively.

B. Deriving $\mathbf{G}(\boldsymbol{\vartheta}_{\beta,\gamma,\lambda})$

The derivation below uses the following matrix identity

$$\text{vec}(\mathbf{ABC}) = (\mathbf{C}^T \otimes \mathbf{A})\text{vec}(\mathbf{B}), \quad (\text{B1})$$

where \mathbf{A} , \mathbf{B} , and \mathbf{C} are matrices of compatible sizes. In addition, we make use of the following relations

$$\text{vec}(\Sigma_\xi) = \mathbf{L}_\xi \boldsymbol{\sigma}_\xi, \quad \text{vec}(\Sigma_\delta) = \mathbf{L}_\delta \boldsymbol{\sigma}_\delta, \quad \text{vec}(\Sigma_{\epsilon_1}) = \mathbf{L}_{\epsilon_1} \boldsymbol{\sigma}_{\epsilon_1}, \quad (\text{B2})$$

$$\text{vec}(\Sigma_{\epsilon_2}) = \mathbf{L}_{\epsilon_2} \boldsymbol{\sigma}_{\epsilon_2}.$$

The model-implied covariance matrix is

$$\Sigma_x(\boldsymbol{\vartheta}) = \begin{pmatrix} \Sigma_{x_1}(\boldsymbol{\vartheta}) & \Sigma_{x_2, x_1}^T(\boldsymbol{\vartheta}) \\ \Sigma_{x_2, x_1}(\boldsymbol{\vartheta}) & \Sigma_{x_2}(\boldsymbol{\vartheta}) \end{pmatrix} \\ = \begin{pmatrix} \mathbf{A}_1 \mathbf{H} (\Gamma \Sigma_\xi \Gamma^T + \Sigma_\delta) \mathbf{H}^T \mathbf{A}_1^T + \Sigma_{\epsilon_1} & \mathbf{A}_1 \mathbf{H} \Gamma \Sigma_\xi \mathbf{A}_2^T + \Sigma_{\epsilon_2, \epsilon_1}^T \\ \mathbf{A}_2 \Sigma_\xi \Gamma^T \mathbf{H}^T \mathbf{A}_1^T + \Sigma_{\epsilon_2, \epsilon_1} & \mathbf{A}_2 \Sigma_\xi \mathbf{A}_2^T + \Sigma_{\epsilon_2} \end{pmatrix} \\ = \begin{pmatrix} \mathbf{A}_1 \mathbf{H} \Gamma \Sigma_\xi \Gamma^T \mathbf{H}^T \mathbf{A}_1^T + \mathbf{A}_1 \mathbf{H} \Sigma_\delta \mathbf{H}^T \mathbf{A}_1^T + \Sigma_{\epsilon_1} & \mathbf{A}_1 \mathbf{H} \Gamma \Sigma_\xi \mathbf{A}_2^T + \Sigma_{\epsilon_2, \epsilon_1}^T \\ \mathbf{A}_2 \Sigma_\xi \Gamma^T \mathbf{H}^T \mathbf{A}_1^T + \Sigma_{\epsilon_2, \epsilon_1} & \mathbf{A}_2 \Sigma_\xi \mathbf{A}_2^T + \Sigma_{\epsilon_2} \end{pmatrix}. \quad (\text{B3})$$

Applying SNLLS, the key is to express the model-implied covariance vector using the form

$$\tilde{\boldsymbol{\sigma}}_x(\boldsymbol{\vartheta}) = \left(\boldsymbol{\sigma}_{x_1}^T(\boldsymbol{\vartheta}) \ \boldsymbol{\sigma}_{x_2}^T(\boldsymbol{\vartheta}) \ \boldsymbol{\sigma}_{x_2, x_1}^T(\boldsymbol{\vartheta}) \right)^T \\ := \mathbf{G}(\boldsymbol{\vartheta}_{\beta,\gamma,\lambda}) \boldsymbol{\sigma}_{\xi, \delta, \epsilon}^T. \quad (\text{B4})$$

To do so, it is necessary to vectorize the individual blocks of (B3). Starting with the block $\Sigma_{x_1}(\boldsymbol{\vartheta})$, we have

$$\boldsymbol{\sigma}_{x_1}(\boldsymbol{\vartheta}) = \mathbf{K}_{x_1}^T \text{vec}(\Sigma_{x_1}(\boldsymbol{\vartheta})) \\ = \mathbf{K}_{x_1}^T \text{vec}(\mathbf{A}_1 \mathbf{H} \Gamma \Sigma_\xi \Gamma^T \mathbf{H}^T \mathbf{A}_1^T) + \mathbf{K}_{x_1}^T \text{vec}(\mathbf{A}_1 \mathbf{H} \Sigma_\delta \mathbf{H}^T \mathbf{A}_1^T) + \mathbf{K}_{x_1}^T \text{vec}(\Sigma_{\epsilon_1}) \\ = \mathbf{K}_{x_1}^T (\mathbf{A}_1 \mathbf{H} \Gamma \otimes \mathbf{A}_1 \mathbf{H} \Gamma) \text{vec}(\Sigma_\xi) + \mathbf{K}_{x_1}^T (\mathbf{A}_1 \mathbf{H} \otimes \mathbf{A}_1 \mathbf{H}) \text{vec}(\Sigma_\delta) + \mathbf{K}_{x_1}^T \text{vec}(\Sigma_{\epsilon_1}). \quad (\text{B5})$$

Using (B1) and (B2), it follows that

$$\boldsymbol{\sigma}_{x_1}(\boldsymbol{\vartheta}) = \mathbf{K}_{x_1}^T (\mathbf{A}_1 \mathbf{H} \Gamma \otimes \mathbf{A}_1 \mathbf{H} \Gamma) \mathbf{L}_\xi \boldsymbol{\sigma}_\xi + \mathbf{K}_{x_1}^T (\mathbf{A}_1 \mathbf{H} \otimes \mathbf{A}_1 \mathbf{H}) \mathbf{L}_\delta \boldsymbol{\sigma}_\delta + \mathbf{K}_{x_1}^T \mathbf{L}_{\epsilon_1} \boldsymbol{\sigma}_{\epsilon_1} \\ = \left(\mathbf{K}_{x_1}^T (\mathbf{A}_1 \mathbf{H} \Gamma \otimes \mathbf{A}_1 \mathbf{H} \Gamma) \mathbf{L}_\xi \ \mathbf{K}_{x_1}^T (\mathbf{A}_1 \mathbf{H} \otimes \mathbf{A}_1 \mathbf{H}) \mathbf{L}_\delta \ \mathbf{K}_{x_1}^T \mathbf{L}_{\epsilon_1} \ \mathbf{0} \ \mathbf{0} \right) \boldsymbol{\sigma}_{\xi, \delta, \epsilon}^T. \quad (\text{B6})$$

Next, we consider the block $\Sigma_{x_2}(\boldsymbol{\vartheta})$. Using the same procedure as before, we have

$$\boldsymbol{\sigma}_{x_2}(\boldsymbol{\vartheta}) = \mathbf{K}_{x_2}^T \text{vec}(\Sigma_{x_2}(\boldsymbol{\vartheta})) \\ = \mathbf{K}_{x_2}^T \text{vec}(\mathbf{A}_2 \Sigma_\xi \mathbf{A}_2^T) + \mathbf{K}_{x_2}^T \text{vec}(\Sigma_{\epsilon_2}) \\ = \mathbf{K}_{x_2}^T (\mathbf{A}_2 \otimes \mathbf{A}_2) \text{vec}(\Sigma_\xi) + \mathbf{K}_{x_2}^T \text{vec}(\Sigma_{\epsilon_2}) \\ = \mathbf{K}_{x_2}^T (\mathbf{A}_2 \otimes \mathbf{A}_2) \mathbf{L}_\xi \boldsymbol{\sigma}_\xi + \mathbf{K}_{x_2}^T \mathbf{L}_{\epsilon_2} \boldsymbol{\sigma}_{\epsilon_2} \\ = \left(\mathbf{K}_{x_2}^T (\mathbf{A}_2 \otimes \mathbf{A}_2) \mathbf{L}_\xi \ \mathbf{0} \ \mathbf{0} \ \mathbf{K}_{x_2}^T \mathbf{L}_{\epsilon_2} \ \mathbf{0} \right) \boldsymbol{\sigma}_{\xi, \delta, \epsilon}^T. \quad (\text{B7})$$

Finally, for the block $\Sigma_{x_2, x_1}(\boldsymbol{\vartheta})$, it follows that

$$\begin{aligned}
 \sigma_{x_2, x_1}(\boldsymbol{\vartheta}) &= \text{vec}(\Sigma_{x_2, x_1}(\boldsymbol{\vartheta})) \\
 &= \text{vec}(A_2 \Sigma_{\xi} \Gamma^T \mathbf{H}^T A_1^T) + \text{vec}(\Sigma_{\epsilon_2, \epsilon_1}) \\
 &= (A_1 \mathbf{H} \Gamma \otimes A_2) \text{vec}(\Sigma_{\xi}) + \text{vec}(\Sigma_{\epsilon_2, \epsilon_1}) \\
 &= (A_1 \mathbf{H} \Gamma \otimes A_2) L_{\xi} \sigma_{\xi} + \sigma_{\epsilon_2, \epsilon_1} \\
 &= ((A_1 \mathbf{H} \Gamma \otimes A_2) L_{\xi} \mathbf{0} \mathbf{0} \mathbf{0} \mathbf{I}) \sigma_{\xi, \delta, \epsilon}.
 \end{aligned}$$

Putting the pieces together, we obtain

$$\begin{aligned}
 & \begin{pmatrix} \sigma_{x_1}(\boldsymbol{\vartheta}) \\ \sigma_{x_2}(\boldsymbol{\vartheta}) \\ \sigma_{x_2, x_1}(\boldsymbol{\vartheta}) \end{pmatrix} \\
 \text{(B8)} \quad &= \begin{pmatrix} K_{x_1}^T (A_1 \mathbf{H} \Gamma \otimes A_1 \mathbf{H} \Gamma) L_{\xi} & K_{x_1}^T (A_1 \mathbf{H} \otimes A_1 \mathbf{H}) L_{\delta} & K_{x_1}^T L_{\epsilon_1} & \mathbf{0} & \mathbf{0} \\ K_{x_2}^T (A_2 \otimes A_2) L_{\xi} & \mathbf{0} & \mathbf{0} & K_{x_2}^T L_{\epsilon_2} & \mathbf{0} \\ (A_1 \mathbf{H} \Gamma \otimes A_2) L_{\xi} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} \end{pmatrix} \\
 & \times \sigma_{\xi, \delta, \epsilon}.
 \end{aligned} \tag{B9}$$