Fast and Reliable Computation of Generalized Synthetic Controls

Martin Becker*
Statistics and Econometrics
Saarland University

Stefan Klößner
Statistics and Econometrics
Saarland University

Preliminary version: January 30, 2017

Abstract

As existing implementations of synthetic control methods suffer from serious weaknesses, we develop new methods for calculating synthetic control units. In particular, we show how to detect and handle important special cases that have not been addressed in the literature yet. We also elaborate on solving the nested optimization associated with the standard case fast and reliably. With the R package **MSCMT**, we provide an open source implementation of the presented methods, which can be applied to generalizations of 'standard' synthetic control methods, too.

Keywords: Synthetic Control Methods; Fast Algorithms; Reliable Computation

JEL Codes: C31, C63, C87

*Correspondence to: Martin Becker, Saarland University, Campus C3 1, 66123 Saarbrücken, Germany. E-mail: martin.becker@mx.uni-saarland.de

1 Introduction

Synthetic control methods (SCM), introduced by Abadie and Gardeazabal (2003) and Abadie et al. (2010), have become an indispensable tool for program evaluation. Recently, numerous studies have appeared that use SCM to analyze the effects of interventions, unforeseen events or structural breaks, e.g., Cavallo et al. (2013), Jinjarak et al. (2013), Kleven et al. (2013), Abadie et al. (2015), Pinotti (2015), Acemoglu et al. (2016), Gobillon and Magnac (2016), to mention just the tip of the iceberg. Very recently, they have also been found to compare well against alternative panel data methods for program evaluation, see Gardeazabal and Vega-Bayo (2016).

Implementations of SCM have been provided by the pioneers of SCM for various software applications, namely Matlab, R (with package **Synth**) and Stata.[1] In a recent replication study, Becker and Klößner (2017a) illustrate that all these implementations fail to find the correct synthetic control unit. However, as we illustrate in this paper, this is not a sole exception: existing implementations of SCM are unreliable in general.

In this paper, we therefore provide helpful results on the theory of the optimization problems that have to be solved when a synthetic control is calculated. On the one hand, these theoretical results enable us to detect important special cases that have been overlooked by the literature. We provide algorithmic solutions for detecting and solving these special cases, leading to exact results for the synthetic control which can be calculated extremely fast. Another benefit of the theoretical results is a potential reduction of the problem's dimension, which increases speed and numerical stability of the nested optimization task associated with the default case. We elaborate on cleverly attacking this demanding and time-consuming search for the synthetic control using numerical optimizers, with special emphasis on speed of computations and reliability of reported results.

All these routines have been implemented in R package **MSCMT** (**M**ultivariate **S**ynthetic **C**ontrol **M**ethod using **T**ime Series), which is open source and publicly available.[2] Furthermore, package **MSCMT** as well as all methods developed in this paper can be applied not only to 'standard' SCM applications, but also to the generalized MSCMT approach of Klößner and Pfeifer (2015), which allows to consider several variables of interest simultaneously and to treat predictor variables as time series.

The remainder of this paper unfolds as follows: in Section 2, we present the standard SCM method as introduced by Abadie and Gardeazabal (2003) and Abadie et al. (2011) as well as its generalization MSCMT of Klößner and Pfeifer (2015), and provide a unifying framework for general synthetic control methods. Section 3 is devoted to results on the theory of SCM optimization problems, detecting and solving special cases, and fast and stable algorithms for computing synthetic controls. In Section 4, we compare our new

[1]See R Core Team (2016), Abadie et al. (2011) and Jens Hainmueller's webpage, `http://web.stanford.edu/~jhain/synthpage.html`.

[2]See Becker and Klößner (2017b), `https://cran.r-project.org/package=MSCMT`.

implementation of synthetic control methods to the already existing ones, with special emphasis on reliability of results, while Section 5 concludes.

2 The Synthetic Control Method

Synthetic control methods have been introduced to the literature by Abadie and Gardeazabal (2003), theory on their (asymptotic) properties has been developed in Abadie et al. (2010), while recently, Gardeazabal and Vega-Bayo (2016) have shown that SCM compares well against the panel data approach to program evaluation developed by Hsiao et al. (2012). Synthetic control methods have also been generalized by Klößner and Pfeifer (2015) to handle multiple variables of interest simultaneously and treat economic predictors as time series (**M**ultivariate **S**ynthetic **C**ontrol **M**ethod using **T**ime Series). In the following, we shortly describe both the 'standard' SCM method of Abadie and Gardeazabal (2003) and the generalized approach by Klößner and Pfeifer (2015), before providing a unifying framework ('General SCM Task') that later sections will build on.

2.1 'Standard' SCM

The synthetic control method as introduced by Abadie and Gardeazabal (2003) and Abadie et al. (2010) aims at producing a synthetic control unit which is compared against the actually treated unit. This synthetic control unit is created as a weighted combination of a collection of J control units, the so-called donor pool. For the synthetic control unit to approximate the treated unit *post treatment*, it is essential that the synthetic control unit comes as close as possible to the treated unit in terms of *pre-treatment* values. This pre-treatment fit is determined not only with respect to the main outcome of interest, but also with respect to a set of so-called (economic) predictors which consists of (economic) variables with explanatory power for the outcome of interest. The standard SCM approach introduces two different kinds of predictors: the first kind is given by m linear combinations of the outcome of interest, Y, in M pre-treatment periods, while the second kind consists of r other covariates with explanatory power for Y. All K predictors (with $K = r + m$) are combined to form a $(K \times 1)$ vector X_1 for the treated unit and a $(K \times J)$ matrix X_0 for all control units.

In one part of the optimization process, called the inner optimization, one aims at finding a linear combination of the columns of X_0 that represents X_1 best, i.e., one searches for a combination of the donor units such that the difference of the predictors' values of the treated unit and the counterfactual becomes as small as possible. The distance metric used to measure this difference is: $\|X_1 - X_0 W\|_V = \sqrt{(X_1 - X_0 W)' V (X_1 - X_0 W)}$, where the weights used to construct the synthetic control unit are denoted by the vector W and the weights of the predictors are given by the nonnegative diagonal matrix V. The latter takes into consideration that not all predictors have the same predictive power for the outcome variable Y.

The inner optimization is then the task of finding, for given predictor weights V, non-negative donor weights W summing up to unity such that[3]

$$\sqrt{(X_1 - X_0W)' \, V \, (X_1 - X_0W)} \overset{W}{\to} \min . \tag{1}$$

The solution to this problem is denoted by $W^*(V)$.

The second part of the optimization, the outer optimization, deals with finding optimal predictor weights V. It usually follows a data-driven approach proposed by Abadie and Gardeazabal (2003) and Abadie et al. (2010): V is chosen among all positive definite and diagonal matrices such that the mean squared prediction error (MSPE) of the outcome variable Y is minimized over the pre-intervention periods.[4] To this end, we denote by Z_1 the M values of Y for the treated unit over the pre-intervention periods, while Z_0 denotes the analogous matrix $M \times J$-matrix for the control units. The outer optimization problem then consists of:

$$(Z_1 - Z_0W^*(V))' \, (Z_1 - Z_0W^*(V)) \overset{V}{\to} \min . \tag{2}$$

2.2 Generalizations of SCM

In Klößner and Pfeifer (2015), the original setup of Abadie and Gardeazabal (2003) and Abadie et al. (2010) has been extended to incorporate more than one dependent variable as well as to treat the predictor data as time series, while maintaining the underlying structure with respect to the donor weights W, the predictor weights V, and the inner and outer optimizations. In particular, Klößner and Pfeifer (2015) extend the equations (1) and (2) for the approximation errors with respect to predictor and outcome data to the following ones:[5]

$$\Delta_X(v_1, \ldots, v_K, W) := \sqrt{\sum_{k=1}^{K} v_k \frac{1}{N_k} \sum_{n=1}^{N_k} \left(X_{k,n,1} - \sum_{j=2}^{J+1} X_{k,n,j} w_j \right)^2} , \tag{3}$$

$$\Delta_Y(W) := \sqrt{\sum_{l=1}^{2} \frac{1}{M_l^{\text{pre}}} \sum_{m=1}^{M_l^{\text{pre}}} \left(Y_{l,m,1} - \sum_{j=2}^{J+1} Y_{l,m,j} w_j \right)^2} . \tag{4}$$

[3]Notice that, prior to any calculations, all predictors are rescaled to unit variance.

[4]In the literature, there exists another approach for determining the predictor weights which is called the 'regression-based' approach. However, it is used quite rarely, the corresponding formulas can be found in Kaul et al. (2016).

[5]Notice that, prior to any calculations, all variables are rescaled such that every dependent variable and every predictor has unit variance.

To explain the formulas (3) and (4), we adopt the notation of Klößner and Pfeifer (2015) and denote the m-th outcome Y of the dependent variable l for unit j by $Y_{l,m,j}$, with $l = 1, 2$ running over the variables of interest and $m = 1, \ldots, M_l^{\text{pre}}$ running over the M_l^{pre} pre-treatment observations of variable l. As above, $j = 1, \ldots, J + 1$ runs through the units, with $j = 1$ denoting the treated unit and $j = 2, \ldots, J + 1$ for the control units. The values of K economic predictors are denoted similarly by $X_{k,n,j}$, with $k = 1, \ldots, K$ running over all economic predictors and $n = 1, \ldots, N_k$ running over the pre-treatment observation times of economic predictor k.

In this paper, we generalize equation (4) a little bit further to

$$\Delta_Y(W) := \sqrt{\sum_{l=1}^{L} \alpha_l \frac{1}{M_l^{\text{pre}}} \sum_{m=1}^{M_l^{\text{pre}}} \beta_{l,m} \left(Y_{l,m,1} - \sum_{j=2}^{J+1} Y_{l,m,j} w_j \right)^2}, \qquad (5)$$

allowing for

- a number L of possibly more than two dependent variables,
- weights $\alpha_1, \ldots, \alpha_L$ describing the importance of the dependent variables, in order to be able to grant more important dependent variables more weight than less important ones with respect to the outer optimization,
- weights $\beta_{l,1}, \ldots, \beta_{l,M_l^{\text{pre}}}$ for the pre-treatment values of the l-th dependent variable, e.g. in order to give more importance to discrepancies close to the point of intervention.

We also slightly generalize equation (3) to

$$\Delta_X(v_1, \ldots, v_K, W) := \sqrt{\sum_{k=1}^{K} v_k \frac{1}{N_k} \sum_{n=1}^{N_k} \gamma_{k,n} \left(X_{k,n,1} - \sum_{j=2}^{J+1} X_{k,n,j} w_j \right)^2}, \qquad (6)$$

with weights $\gamma_{k,1}, \ldots, \gamma_{k,N_k}$ analogously to the weights β above, allowing the inner optimization to be tailored such that the fit of the k-th independent variable at times close to the point of intervention becomes more important.

Consistent with the usual SCM approach, the inner optimization aims, for given $v_1, \ldots, v_K$, at finding $W^*(v_1, \ldots, v_K) = (w_2^*, \ldots, w_{J+1}^*)'$ such that the inner objective function (6) is minimized with respect to W.[6] The outer optimization in turn minimizes $\Delta_Y(W^*(v_1, \ldots, v_K))$ or equivalently $\Delta_Y^2(W^*(v_1, \ldots, v_K))$, the (root) mean square error of the dependent variables' approximation.

[6]Obviously, instead of minimizing $\Delta_X(v_1, \ldots, v_K, W)$, the root mean square error of the predictors' approximation as given in (6), one can equivalently minimize $\Delta_X^2(v_1, \ldots, v_K, W)$, the mean square error of the approximation.

2.3 General SCM Problem Formulation

All the SCM variants discussed above can be translated into a common, very general structure which will be called 'general SCM problem formulation' in the sequel. For instance, by defining the $M \times J$-matrix $\widetilde{Z} := Z_0 - Z_1 \mathbb{1}'$,[7] the $K \times J$-matrix $\widetilde{X} := X_0 - X_1 \mathbb{1}'$, and the mapping V by

$$V(v_1, \ldots, v_K) := \begin{pmatrix} v_1 & 0 & \ldots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \ldots & 0 & v_K \end{pmatrix} \in \mathbb{R}^{K \times K},$$

the standard SCM approach given by equations (1) and (2) can be described as the following nested optimization task:

- in the inner optimization, find, for given $v_1, \ldots, v_K$, $W^*(v_1, \ldots, v_K)$ as the minimizer of $w' \widetilde{X}' V(v_1, \ldots, v_K) \widetilde{X} w$ over all vectors w whose non-negative entries sum to unity,
- in the outer optimization, minimize $W^*(v_1, \ldots, v_K)' \widetilde{Z}' \widetilde{Z} W^*(v_1, \ldots, v_K)$ with respect to $v_1, \ldots, v_K$.

Similarly, the very general SCM method introduced in the previous section might be brought into analogous form by denoting $M := \sum_{l=1}^{L} M_l^{\text{pre}}$ and $N := \sum_{k=1}^{K} N_k$ as well as defining the $M \times J$-matrix

$$\widetilde{Z} := \left(\begin{array}{ccc} \sqrt{\frac{\alpha_1 \beta_{1,1}}{M_1^{\text{pre}}}} \left(Y_{1,1,2} - Y_{1,1,1}\right) & \ldots & \sqrt{\frac{\alpha_1 \beta_{1,1}}{M_1^{\text{pre}}}} \left(Y_{1,1,J+1} - Y_{1,1,1}\right) \\ \vdots & \vdots & \vdots \\ \sqrt{\frac{\alpha_1 \beta_{1,M_1^{\text{pre}}}}{M_1^{\text{pre}}}} \left(Y_{1,M_1^{\text{pre}},2} - Y_{1,M_1^{\text{pre}},1}\right) & \ldots & \sqrt{\frac{\alpha_1 \beta_{1,M_1^{\text{pre}}}}{M_1^{\text{pre}}}} \left(Y_{1,M_1^{\text{pre}},J+1} - Y_{1,M_1^{\text{pre}},1}\right) \\ \hdashline \vdots & \vdots & \vdots \\ \hdashline \sqrt{\frac{\alpha_L \beta_{L,1}}{M_L^{\text{pre}}}} \left(Y_{L,1,2} - Y_{L,1,1}\right) & \ldots & \sqrt{\frac{\alpha_L \beta_{L,1}}{M_L^{\text{pre}}}} \left(Y_{L,1,J+1} - Y_{L,1,1}\right) \\ \vdots & \vdots & \vdots \\ \sqrt{\frac{\alpha_L \beta_{L,M_L^{\text{pre}}}}{M_L^{\text{pre}}}} \left(Y_{L,M_L^{\text{pre}},2} - Y_{L,M_L^{\text{pre}},1}\right) & \ldots & \sqrt{\frac{\alpha_L \beta_{L,M_L^{\text{pre}}}}{M_L^{\text{pre}}}} \left(Y_{L,M_L^{\text{pre}},J+1} - Y_{L,M_L^{\text{pre}},1}\right) \end{array}\right),$$

[7] $\mathbb{1}$ denotes the vector of ones.

the $N \times J$-matrix

$$\widetilde{X} := \left(\begin{array}{ccc}
\sqrt{\frac{\gamma_{1,1}}{N_1}}\left(X_{1,1,2} - X_{1,1,1}\right) & \dots & \sqrt{\frac{\gamma_{1,1}}{N_1}}\left(X_{1,1,J+1} - X_{1,1,1}\right) \\
\vdots & \vdots & \vdots \\
\sqrt{\frac{\gamma_{1,N_1}}{N_1}}\left(X_{1,N_1,2} - X_{1,N_1,1}\right) & \dots & \sqrt{\frac{\gamma_{1,N_1}}{N_1}}\left(X_{1,N_1,J+1} - X_{1,N_1,1}\right) \\
\hdashline
\vdots & \vdots & \vdots \\
\hdashline
\sqrt{\frac{\gamma_{K,1}}{N_K}}\left(X_{K,1,2} - X_{K,1,1}\right) & \dots & \sqrt{\frac{\gamma_{K,1}}{N_K}}\left(X_{K,1,J+1} - X_{K,1,1}\right) \\
\vdots & \vdots & \vdots \\
\sqrt{\frac{\gamma_{K,N_K}}{N_K}}\left(X_{K,N_K,2} - X_{K,N_K,1}\right) & \dots & \sqrt{\frac{\gamma_{K,N_K}}{N_K}}\left(X_{K,N_K,J+1} - X_{K,N_K,1}\right)
\end{array}\right),$$

and the linear mapping

$$V(v_1, \dots, v_K) := \begin{pmatrix} v_1 I_{N_1} & 0 & \dots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & v_K I_{N_K} \end{pmatrix},$$

with I_{N_k} denoting the N_k-dimensional identity matrix.

In the sequel, we will therefore, both with respect to theory and algorithms, consider the following general SCM optimization task.

General SCM Task: For the following input variables,

- an $N \times J$-matrix $\widetilde{X}$ (used in the inner optimization),
- an $M \times J$-matrix $\widetilde{Z}$ (used in the outer optimization),
- a linear function V mappping $v_1, \dots, v_K$ to an $N \times N$ diagonal matrix,

determine $v_1^*, \dots, v_K^*$ such that

$$W^*(v_1, \dots, v_K)\widetilde{Z}'\widetilde{Z}W^*(v_1, \dots, v_K) \tag{7}$$

becomes as small as possible (outer optimization), with $W^*(v_1, \dots, v_K)$ being defined as

$$W^*(v_1, \dots, v_K) = \underset{\substack{w \geq 0 \\ \mathbb{1}'w=1}}{\arg\min}\, w'\widetilde{X}'V(v_1, \dots, v_K)\widetilde{X}w, \tag{8}$$

i.e. as the minimizer of the inner objective function.[8]

[8]If $\arg\min_x F(x)$ is a singleton $\{x^*\}$, $\arg\min_x F(x)$ shall denote its single element x^* throughout the paper.

3 Solving Generalized SCM Problems

In the following, we will discuss how finding the solution to a generalized SCM task can be decomposed into several steps. In particular, we show how to detect and handle important special cases, render the optimization problem mathematically sound, tackle the nested optimization task associated with the standard case, and provide a stylized algorithm for solving generalized SCM tasks.

3.1 Dimension Reduction and Special Cases

In order to detect important special cases as well as to potentially speed up the inner optimization, we introduce the notion of 'sunny' donors. To this end, we take a closer look at the structure of the inner optimization task, which is, for given $v_1, \ldots, v_K$, to miminize the quadratic form $w'\widetilde{X}'V(v_1, \ldots, v_K)\widetilde{X}w = ||V(v_1, \ldots, v_K)^{\frac{1}{2}}\widetilde{X}w||^2$ over all non-negative weights $w \in \mathbb{R}^J$ whose components sum up to unity. The inner objective function thus aims at minimizing an appropriately weighted sum of the squared differences $\widetilde{X}w$, where the diagonal matrix $V(v_1, \ldots, v_K)^{\frac{1}{2}}$ rescales the vector of differences according to the predictor weights given by $v_1, \ldots, v_K$. Notice that every column of $\widetilde{X}$ corresponds to the data of some donor, or, more precisely, to the difference of that donor's data to the treated unit's data. Additionally, $\widetilde{X}w$ is nothing else than a convex combination of the columns of $\widetilde{X}$, measuring the difference between synthetic and treated unit with respect to the economic predictors. We therefore introduce the following notion of so-called 'sunny' and 'shady' donors: denoting by $\mathbf{H} := \mathrm{conv}(\{\widetilde{x}_1, \ldots, \widetilde{x}_J\}) = \left\{ \sum_{j=1}^J w_j\widetilde{x}_j \,\middle|\, (\forall j : w_j \geq 0), \sum_{j=1}^J w_j = 1\right\}$ the convex hull of the columns of $\widetilde{X}$, we call $\widetilde{x}_j$ $(j = 1, \ldots, J)$ **sunny** (with respect to H or with respect to $\widetilde{x}_1, \ldots, \widetilde{x}_J$) if a beam of light emanating at the origin can reach $\widetilde{x}_j$ without crossing H first, i.e. if $\widetilde{x}_j$ lies on the sunny side of H if the sun is located at the origin. Formalizing this leads to the following definition:

Definition 1. *Given a set $\{\widetilde{x}_1, \ldots, \widetilde{x}_J\}$ of points in N-dimensional space with convex hull $\mathbf{H} := \mathrm{conv}(\{\widetilde{x}_1, \ldots, \widetilde{x}_J\})$, we call $\widetilde{x}_j$ $(j = 1, \ldots, J)$* **shady** *(with respect to H or with respect to $\widetilde{x}_1, \ldots, \widetilde{x}_J$) if there exists $0 < \alpha < 1$ such that $\alpha\widetilde{x}_j \in H$. We call $\widetilde{x}_j$* **sunny** *(with respect to H or with respect to $\widetilde{x}_1, \ldots, \widetilde{x}_J$) if it is not shady.*

Calling the j-th donor sunny if $\widetilde{x}_j$ is sunny, it is easy to check whether a donor is sunny or not. One simply solves the linear program

$$\min \alpha \text{ subject to } \alpha\widetilde{x}_j = \sum_{\widetilde{j}=1}^{J} w_{\widetilde{j}}\widetilde{x}_{\widetilde{j}},\ \alpha \geq 0,\ w_1 \geq 0, \ldots, w_J \geq 0,\ \sum_{\widetilde{j}=1}^{J} w_{\widetilde{j}} = 1 \qquad (9)$$

to get optimal values $\alpha^*, w_1^*, \ldots, w_J^*$:[9] if $\alpha^* = 1$, then the corresponding donor is sunny,

[9]Note that by choosing $\alpha = 1$, $w_j = 1$, and $w_{\widetilde{j}} = 0$ for $\widetilde{j} \neq j$, there always exists an admissible solution

while it is shady whenever $\alpha^* < 1$.

One important special case neglected by the literature, but deserving particular attention, is given when there are no sunny donors at all. The following proposition shows that this happens if and only if $0 \in H$, i.e. if and only if there exists a synthetic control with donor weights w such that the approximation error $\widetilde{X}w$ vanishes.

Proposition 1. $0 \in H$ *if and only if no* $\widetilde{x}_j$ *is sunny.*

Proof. '$\Rightarrow$' For any $\widetilde{x}_j$ and $0 < \alpha < 1$, the fact that $0 \in H$ and $\widetilde{x}_j \in H$ implies that $H \ni \alpha\widetilde{x}_j + (1-\alpha)\,0 = \alpha\widetilde{x}_j$, thus $\widetilde{x}_j$ is shady for all j.

'$\Leftarrow$' First, we define $\alpha_j^* := \inf(\{\alpha : 0 < \alpha \leq 1 \wedge \alpha\widetilde{x}_j \in H\})$. As no $\widetilde{x}_j$ is sunny, we have $\alpha_j^* < 1$ for all $j = 1, \ldots, J$. Furthermore, because H is closed, $\alpha_j^*\widetilde{x}_j \in H$ for all j. Therefore, the proof is complete as soon as we know that $\alpha_j^* = 0$ for at least one j. To proceed, we thus assume that $\alpha_j^* > 0$ for $j = 1, \ldots, J$. As $\alpha_j^*\widetilde{x}_j \in H$ for all j, there exist $w_{\widetilde{j}}^{(j)}$, non-negative for all $j, \widetilde{j} = 1, \ldots, J$, with $\sum_{\widetilde{j}=1}^{J} w_{\widetilde{j}}^{(j)} = 1$ and $\alpha_j^*\widetilde{x}_j = \sum_{\widetilde{j}=1}^{J} w_{\widetilde{j}}^{(j)}\widetilde{x}_{\widetilde{j}}$ for all $j = 1, \ldots, J$, entailing $\widetilde{x}_j = \frac{1}{\alpha_j^*}\sum_{\widetilde{j}=1}^{J} w_{\widetilde{j}}^{(j)}\widetilde{x}_{\widetilde{j}}$. From this, we find

$$\alpha_1^*\widetilde{x}_1 = \sum_{j=1}^{J} w_j^{(1)}\widetilde{x}_j = \sum_{j=1}^{J} w_j^{(1)}\frac{1}{\alpha_j^*}\sum_{\widetilde{j}=1}^{J} w_{\widetilde{j}}^{(j)}\widetilde{x}_{\widetilde{j}} = \sum_{\widetilde{j}=1}^{J}\left(\sum_{j=1}^{J}\frac{w_j^{(1)}}{\alpha_j^*}w_{\widetilde{j}}^{(j)}\right)\widetilde{x}_{\widetilde{j}} = \sum_{\widetilde{j}=1}^{J}\tau_{\widetilde{j}}\widetilde{x}_{\widetilde{j}},$$

with $\tau_{\widetilde{j}} := \sum_{j=1}^{J}\frac{w_j^{(1)}}{\alpha_j^*}w_{\widetilde{j}}^{(j)}$ for all $\widetilde{j} = 1, \ldots, J$. For $\tau := \sum_{\widetilde{j}=1}^{J}\tau_{\widetilde{j}}$, we have

$$\tau = \sum_{\widetilde{j}=1}^{J}\sum_{j=1}^{J}\frac{w_j^{(1)}}{\alpha_j^*}w_{\widetilde{j}}^{(j)} = \sum_{j=1}^{J}\sum_{\widetilde{j}=1}^{J}\frac{w_j^{(1)}}{\alpha_j^*}w_{\widetilde{j}}^{(j)} = \sum_{j=1}^{J}\frac{w_j^{(1)}}{\alpha_j^*} > \sum_{j=1}^{J} w_j^{(1)} = 1,$$

with the strict inequality holding because $\alpha_j^* < 1$ as well as $w_j^{(1)} \geq 0$ for all j, and $w_j^{(1)} > 0$ for at least one j. Overall, we thus have $\frac{\alpha_1^*}{\tau}\widetilde{x}_1 = \sum_{\widetilde{j}=1}^{J}\frac{\tau_{\widetilde{j}}}{\tau}\widetilde{x}_{\widetilde{j}}$, i.e., $\frac{\alpha_1^*}{\tau}\widetilde{x}_1 \in H$. The minimality of α_1^* now implies $\frac{\alpha_1^*}{\tau} \geq \alpha_1^*$ and therefore $\tau \leq 1$, so that we arrive at a contradiction. Thus, at least one of the α_j^* must be zero, concluding the proof. □

Proposition 1 above shows that there exist no sunny donors if and only if there exist donor weights w such that $\widetilde{X}w = 0$, i.e. donor weights such that with respect to the predictors, the treated unit is perfectly equal to the synthetic control given by the donor weights w. In this case, the result of the inner optimization does not depend on $v_1, \ldots, v_K$, as all w with $\widetilde{X}w = 0$ are minimizers of the inner objective function. In particular, it may be the case that there exist different vectors of donor weights with such a perfect predictor fit. We therefore treat this special case of 'no sunny donors' by searching

for the linear program (9).

among all those vectors of donor weights for one that is optimal with respect to the outer objective function, i.e. for the one with the best fit with respect to the outcome data: mathematically, we thus look for non-negative weights w summing to unity and minimizing $w'\widetilde{Z}'\widetilde{Z}w$ subject to $\widetilde{X}w = 0$, i.e., we solve the following quadratic program:

$$\min_{\substack{w \geq 0 \\ \mathbb{1}'w=1,\ \widetilde{X}w=0}} \quad w'\widetilde{Z}'\widetilde{Z}w. \tag{10}$$

For the 'standard' case with at least one sunny donor, Proposition 2 below shows that if w^* is a minimizer of the inner objective function with $(w^*)'\widetilde{X}'V(v_1,\ldots,v_K)\widetilde{X}w^* > 0$, then w^*_j can only be positive for sunny donors $\widetilde{x}_j$. This result is very useful for reducing the dimension of the inner optimization task, because we can safely neglect shady donors and consider only the subset of sunny donors, corresponding to keeping only those columns of $\widetilde{X}$ that belong to sunny donors, and ignoring the ones belonging to shady donors.

Proposition 2. *Let V be a diagonal matrix with positive entries and $\widetilde{x}^* := \sum_{j=1}^J w^*_j \widetilde{x}_j$ an optimizer of $\widetilde{x}'V\widetilde{x} \to \min$ in H, for which $(\widetilde{x}^*)'V\widetilde{x}^* > 0$. Then $w^*_j = 0$ for all shady donors $\widetilde{x}_j$.*

Proof. Let j be such that $\widetilde{x}_j$ is not sunny. Then there exist $w_1^{(j)},\ldots,w_J^{(j)} \geq 0$ and $0 < \alpha < 1$ such that $\sum_{\widetilde{j}=1}^J w_{\widetilde{j}}^{(j)} = 1$ and $\sum_{\widetilde{j}=1}^J w_{\widetilde{j}}^{(j)}\widetilde{x}_{\widetilde{j}} = \alpha\widetilde{x}_j$. Using these, we define

$$\widetilde{w}_{\widetilde{j}}^{(j)} := \begin{cases} \frac{w^*_{\widetilde{j}} + w^*_j \frac{1}{\alpha} w_{\widetilde{j}}^{(j)}}{1+w^*_j\left(\frac{1}{\alpha}-1\right)} & : \ \widetilde{j} \neq j \\ \frac{w^*_j \frac{1}{\alpha} w_j^{(j)}}{1+w^*_j\left(\frac{1}{\alpha}-1\right)} & : \ \widetilde{j} = j \end{cases},$$

which are non-negative and sum to unity for all $j = 1,\ldots,J$: $\sum_{\widetilde{j}=1}^J \widetilde{w}_{\widetilde{j}}^{(j)} = 1$. We then have $\frac{1}{1+w^*_j\left(\frac{1}{\alpha}-1\right)}\widetilde{x}^* = \sum_{\widetilde{j}=1}^J \widetilde{w}_{\widetilde{j}}^{(j)}\widetilde{x}_{\widetilde{j}}$, because $\frac{1}{\alpha}\sum_{\widetilde{j}=1}^J w_{\widetilde{j}}^{(j)}\widetilde{x}_{\widetilde{j}} = \widetilde{x}_j$. Therefore, $\frac{1}{1+w^*_j\left(\frac{1}{\alpha}-1\right)}\widetilde{x}^* = \sum_{\widetilde{j}=1}^J \widetilde{w}_{\widetilde{j}}^{(j)}\widetilde{x}_{\widetilde{j}}$ lies in H, for which $\left(\frac{1}{1+w^*_j\left(\frac{1}{\alpha}-1\right)}\widetilde{x}^*\right)' V \frac{1}{1+w^*_j\left(\frac{1}{\alpha}-1\right)}\widetilde{x}^* = \frac{1}{\left(1+w^*_j\left(\frac{1}{\alpha}-1\right)\right)^2}\left(\widetilde{x}^*\right)' V\widetilde{x}^*$. Thus, as $\widetilde{x}^*$ minimizes $\widetilde{x}'V\widetilde{x}$ in H, $1 + w^*_j\left(\frac{1}{\alpha}-1\right)$ must not exceed 1, which entails $w^*_j = 0$. □

As stated above, the advantage of Proposition 2 lies in the fact that it allows to reduce the dimension of the inner optimization task: all shady donors can be neglected and the corresponding columns of $\widetilde{X}$ and $\widetilde{Z}$ can be dropped.[10] Additionally, it can also happen that the set of sunny donors is a singleton: in this case, regardless of the predictor weights $v_1,\ldots,v_K$, the solution of the inner optimization task will always be given by the only sunny donor. Then, it is unnecessary to run the outer optimization task, as the

[10]For ease of notation, we do not introduce additional notation for $\widetilde{X}$ and $\widetilde{Z}$ after the dropping of corresponding columns, denoting the dimension-reduced matrices still as $\widetilde{X}$ and $\widetilde{Z}$.

treated unit will be synthesized by the single sunny donor, while $v_1, \dots, v_K$ can be chosen arbitrarily.

All in all, we discern three cases: first, if there are no sunny donors, we search donor weights w that minimize the outer objective function subject to a perfect fit with respect to the predictors, i.e. we look for non-negative donor weights w summing up to unity that minimize $w'\widetilde{Z}'\widetilde{Z}w$ subject to $\widetilde{X}w = 0$. Thus, a quadratic optimization problem with linear restrictions has to be solved, while there is no need to search for optimal predictor weights. In the second case, when there is only one sunny donor, there is again no need to optimize over predictor weights, as the solution of the inner optimization task is always given be the uniquely sunny donor, irrespective of any predictor weights. In the third and last case, the outer optimization task has to be carried out, but we potentially gain a significant amount of computing speed by considering only sunny donors and dropping all columns of $\widetilde{X}$ that belong to shady donors.

3.2 The Domain of the Outer Optimization

The domain of the outer optimization is closely connected to the domain of the input variables $v_1, \dots, v_K$ of the inner optimizer. These input variables correspond, potentially apart from simple transformations, to the weights of the K predictors. Abadie and Gardeazabal (2003) require these predictor weights to be nonnegative, however, there is obviously no unique minimizer of the inner optimization when all predictor weights vanish. So, in a first step, the zero vector has to be excluded.

Obviously, the function $W^*(v_1, \dots, v_K) = W^*(v)$ (with $v = (v_1, \dots, v_K)$), which maps the input variables $v_1, \dots, v_K$ on the solution of the inner optimization, is (positive) homogeneous of order zero, i.e., for all $\alpha > 0$ and $v_1 \geq 0, \dots, v_K \geq 0$ with $v \neq 0$, the identity

$$W^*(v_1, \dots, v_K) = W^*(\alpha v_1, \dots, \alpha v_K) \tag{11}$$

holds. In other words, the outer optimizer's objective function, when restricted to rays $\{(\alpha v_1, \dots, \alpha v_K) | \alpha > 0\}$, is constant for every $(v_1, \dots, v_K)$ with $v_1 \geq 0, \dots, v_K \geq 0$ and $v \neq 0$. Therefore, the (nonnegative) input variables $v_1, \dots, v_K$ of the inner optimizer can be normalized arbitrarily. Without loss of generality, we assume $\max\{v_1, \dots, v_K\} \equiv 1$, which is easily obtained by the mapping

$$(v_1, \dots, v_K) \mapsto \left(\frac{v_1}{\max\{v_1, \dots, v_K\}}, \dots, \frac{v_K}{\max\{v_1, \dots, v_K\}} \right) \tag{12}$$

considering that $\max\{v_1, \dots, v_K\} > 0$ because $v \neq 0$.

As we experienced, it is not sufficient to require that at least one predictor weight is different from zero, because W^* and, as a consequence, the outer objective function are often discontinuous around predictor weights that are not entirely positive.[11] Thus, for W^* and the outer objective function to be continuous on their domain of definition, $v_1, \ldots, v_K$ must be bounded away from zero. Additionally, the ratio $\frac{\min(v_1,\ldots,v_K)}{\max(v_1,\ldots,v_K)}$ should not become too small, as this would lead to arbitrarily ill-conditioned matrices $V(v_1, \ldots, v_K)^{\frac{1}{2}}\widetilde{X}$ appearing in the inner optimization. Thus, both in order to work with continuous functions and to guarantee numerical stability of the inner optimization, we restrict the ratio $\frac{\min(v_1,\ldots,v_K)}{\max(v_1,\ldots,v_K)}$ not to fall below some lower bound lb.[12] We propose $lb = 10^{-8}$ as a reasonable choice for this lower bound in order to regularly obtain feasible predictor weights $v_1, \ldots, v_K$.

3.3 Feasibility of the Unrestricted Outer Optimum and Finding Predictor Weights

Finding a synthetic control is not only a procedure consisting of an inner and an outer optimization, it also pursues two goals: the synthetic control unit should resemble the treated unit, prior to the treatment, both with respect to the predictors (inner objective function $w'\widetilde{X}'V(v_1, \ldots, v_K)\widetilde{X}w$) and the outcome(s) (outer objective function $W^*(v_1, \ldots, v_K)'\widetilde{Z}'\widetilde{Z}W(v_1, \ldots, v_K)$). Obviously, the outer objective function can not fall below $w'_{\text{outer}}\widetilde{Z}'\widetilde{Z}w_{\text{outer}}$, where $\widetilde{Z}$ has not (yet) been restricted to sunny donors and

$$w_{\text{outer}} := \mathop{\arg\min}_{\substack{w \geq 0 \\ \mathbb{1}'w=1}} w'\widetilde{Z}'\widetilde{Z}w \tag{13}$$

denotes the so-called **unrestricted outer optimum**. Ideally, we would like optimal predictor weights $v_1^*, \ldots, v_K^*$ to be such that $W^*(v_1^*, \ldots, v_K^*) = w_{\text{outer}}$, a case for which the inner and outer objective function are not at odds with each other: in this case, we call the unrestricted outer optimum w_{outer} feasible with respect to the inner minimization. As computing w_{outer} is rather straightforward, it would be very helpful if there was also an easy way of checking whether w_{outer} is feasible. The key to construct such a check is the following proposition.

Proposition 3. *Let B be a symmetric, positive semi-definite J-by-J matrix. A J-dimensional vector w^* with $w_1^*, \ldots, w_J^* \geq 0$ and $\sum_{j=1}^J w_j^* = 1$ is a minimizer of $w'Bw$ in $\{w = (w_1, \ldots, w_J)' : w_1, \ldots, w_J \geq 0, \sum_{j=1}^J w_j = 1\}$ if and only if $(Bw^*)_j \geq (w^*)'Bw^*$ for $j = 1, \ldots, J$. In this case, we additionally have $(Bw^*)_j = (w^*)'Bw^*$ for all j with $w_j^* > 0$.*

Proof. Since B is symmetric and positive semi-definite, the Karush-Kuhn-Tucker condi-

[11]For a corresponding example, see the web appendix.

[12]Below, we will introduce methods to check whether this restriction was actually binding or not.

tions

$$Bw - t + \mathbb{1}u = 0 \tag{14}$$

$$t'w = 0 \tag{15}$$

with $t = (t_1, \dots, t_J) \geq 0$, $u \in \mathbb{R}$, are necessary and sufficient.[13] Multiplying equation (14) from left by w' yields $u = -w'Bw$, which completes the proof. $\square$

Applying Proposition 3 to $B := \widetilde{X}'V(v_1, \dots, v_K)\widetilde{X}$ shows that w minimizes the inner objective function for given $v_1, \dots, v_K$ if and only if no component of $\widetilde{X}'V(v_1, \dots, v_K)\widetilde{X}w$ falls below $w'\widetilde{X}'V(v_1, \dots, v_K)\widetilde{X}w$, with equality for all donors j whose donor weights w_j are positive. Put differently, the above conditions allow to check whether a given vector w of donor weights is feasible, i.e. whether there exist predictor weights $v_1, \dots, v_K$ such that $w = W^*(v_1, \dots, v_K)$. Moreover, checking this is an easy task, because $\widetilde{X}'V(v_1, \dots, v_K)\widetilde{X}w$ as well as $w'\widetilde{X}'V(v_1, \dots, v_K)\widetilde{X}w$ inherit from V the property that they are linear functions of $v_1, \dots, v_K$. Therefore, checking whether a given vector w of donor weights is feasible can be done by checking whether the following linear program in terms of $v_1, \dots, v_K$ admits a feasible solution:

$$\min \sum_{k=1}^{K} c_k v_k \text{ subject to } \begin{cases} \sum_{k=1}^{K} v_k (e_j - w)' B_k w \geq 0 \text{ for all } j = 1, \dots, J, \\ v_1 \geq lb, \dots, v_K \geq lb, \\ v_1 \leq 1, \dots, v_K \leq 1, \end{cases} \tag{16}$$

where $c_1, \dots, c_K$ are arbitrary numbers, e_j denotes the j-th unit vector in $\mathbb{R}^J$, and $B_1 := \widetilde{X}'V(1, 0, \dots, 0)\widetilde{X}, \dots, B_K := \widetilde{X}'V(0, \dots, 0, 1)\widetilde{X}$.

Thus, by putting $w := w_{\text{outer}}$ in (16), it is possible to check whether the unrestricted outer optimum, w_{outer}, is feasible.[14] If so, w_{outer} are optimal weights for synthesizing, and the synthetization task is solved.[15] If not, the inner and outer objective functions constitute conflicting goals, and the nested optimization task consisting of the outer optimization described in equation (7) and the inner optimization described in equation (8) has to be solved.

3.4 The Choice of the Inner Optimizer

Calculating the objective function for the outer optimization requires solving the inner optimization. Thus, in order to make the results of the outer optimization trustworthy, the

[13]See, e.g., (Wolfe, 1959, Theorem 2).

[14] The linear program (16) can easily be modified to check whether the lower bound lb is actually binding: to this end, one can introduce an additional variable $\underline{v}$, change the objective function to maximizing $\underline{v}$, and add the restrictions $\underline{v} \leq v_1, \dots, \underline{v} \leq v_K$. Then, any optimal $\underline{v}^*$ will automatically be chosen as the minimum of $v_1, \dots, v_K$, and the lower bound lb is binding if and only if $\underline{v}^* = lb$.

[15]An example for a feasible unrestricted outer optimum can be found in Becker and Klößner (2017a).

solutions of the inner optimization must be very reliable. Even small errors in the results of the inner optimization may fool the outer optimizer[16], while large errors may have critical impact on the outer optimizer's performance. Furthermore, the inner optimization has to be done once for every evaluation of the objective function of the outer optimization, which typically leads to a huge number of invocations of the inner optimization. This calls for an efficient inner optimization procedure which is not only reliable, but also fast.

Obviously, the inner optimization problem (8) is a quadratic program. There are many different algorithms for quadratic programming, some of which already have implementations in R. Package **Synth**, e.g., makes use of function `ipop` contained in package **kernlab** and, alternatively, function `LowRankQP` contained in package **LowRankQP**.[17] As we experienced, these algorithms and their implementations, as well as some further methods for quadratic programming we have tested, regularly produce (more or less considerably) suboptimal results or even errors.[18]

The key to our solution to this problem is to exploit the well-known fact that problem (8) is a member of a particular subclass of quadratic programs by using its equivalent notation

$$W^*(v_1,\ldots,v_K) = \underset{\substack{w\geq 0\\ \mathbb{1}'w=1}}{\arg\min} ||V(v_1,\ldots,v_K)^{\frac{1}{2}}\widetilde{X}w||^2 \tag{8'}$$

as a nonnegatively constrained linear least squares problem with linear equality constraints.[19]

An algorithm (called `WNNLS`) for this particular class of problems has been introduced in Haskell and Hanson (1981), a corresponding Fortran implementation was presented in Hanson and Haskell (1982). Fortunately, the simple but tedious adaptions necessary to use this implementation with the R-to-Fortran interface have already been done in package **limSolve**[20]. Extensive tests have shown that `WNNLS` is a very reliable and very fast inner optimizer for the nested optimization problem to be solved.

We illustrate the excellent performance of `WNNLS` compared to `ipop` and `LowRankQP` with a benchmark based on the first empirical application discussed later in Section 4. For this purpose, we implemented a special variant for solving the inner optimization task: for every vector of predictor weights considered during the outer optimization, we let all three optimizers (`ipop`, `LowRankQP`, `WNNLS`) provide solutions of the inner optimization[21] and stay with the best solution, but record the inner objective function values given by all optimizers. To obtain information about the accuracy of the inner optimizers, we

[16]Using outer optimizers suited for noisy objective functions may remedy or at least mitigate this effect.

[17]See Karatzoglou et al. (2004) and Ormerod and Wand (2014).

[18]See the corresponding section of the web appendix and the benchmark in Section 4 below.

[19]Note that the optimization problems (10) and (13) belong to this particular subclass as well, which enables the application of the `WNNLS` algorithm proposed below for problems (10) and (13), too.

[20]See Soetaert et al. (2009). We are deeply grateful to Karline Soetaert for the permission to include this adapted code in our reference implementation, the R package **MSCMT**.

[21]We used the default parameters in function `synth` of R package **Synth** for `ipop` and `LowRankQP`.

calculated the relative increase (compared to the minimum) of the three alternative loss function values for all 313573 invocations of the inner optimizer. `WNNLS`'s superiority in terms of accuracy is striking, as `WNNLS` performed best in all invocations[22]. Figure 1 illustrates the relative increases for `ipop` and `LowRankQP` using a density plot with a logarithmically scaled abscissa: `LowRankQP` is perfoming second best with a mean relative increase of 1.978×10^{-8} and a maximal relative increase of 1.635×10^{-4}, while `ipop` results in a mean relative increase of 0.006333 and a maximal relative increase of 15.45.

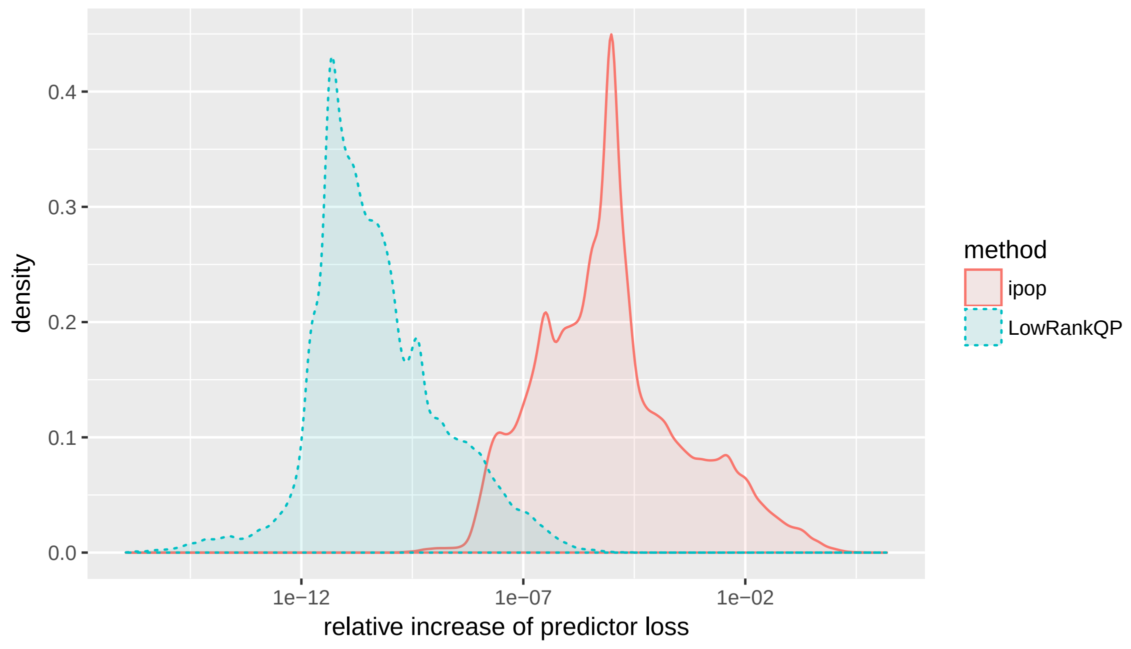

Figure 1: Comparison of relative increases compared to the best predictor loss for `ipop` and `LowRankQP`.

To obtain information about the speed of the inner optimizers, we measured the time required for the evaluation of the inner optimization, separately for all three optimizers, for all recorded predictor weights.[23] Table 1 summarizes the results of the measurement of the CPU time consumption. Again, `WNNLS`'s superiority is obvious, as it is about 21 times faster than `LowRankQP` and about 350 times faster than `ipop`. All in all, `WNNLS` clearly dominates `LowRankQP`, which itself outperforms `ipop`, both in terms of accuracy and speed. `WNNLS` is thus the method of choice for solving the inner optimization problem.

[22]In 313557 out of 313573 optimization tasks, `WNNLS` provided the best solution. In the remaining 16 optimization tasks, `WNNLS` was 'only' best up to machine precision with a maximal relative increase (compared to the result of `LowRankQP`) of 3.476×10^{-16}. Using `ipop` even resulted in an error for 146 out of 313573 optimization tasks.

[23]We used a single core of an Intel® i7 860@2.80GHz.

Inner Optimizer	`ipop`	`LowRankQP`	`WNNLS`
Computation time in seconds (for 313573 evaluations)	1746.97	106.13	4.95
Evaluations per second	179	2955	63348

Table 1: Evaluation speed of the different inner optimizers.

3.5 Details on the Outer Optimization

Even if the inner optimization problem is solved reliably, the outer optimization remains challenging, because the outer objective function typically has many local optima. Therefore, using a simple local optimizer with a single set of starting values is not a promising strategy when looking for the global optimum.

Another difficulty, which we will address first, is the domain (also referred to as the search space) of the outer optimizer and its relation to the input of the inner optimizer. At first sight, the search space of the outer optimizer seems to be identical to the domain of feasible values for the inner optimizer's input variables $v_1, \dots, v_K$. However, since it is possible (and common practice) to transform the elements of the search space of an optimizer before evaluating the objective function, this identity is only optional.

In Subsection 3.2 above, we already proposed to restrict the feasible predictor weights for the inner optimization problem to the set

$$\mathcal{V} := \big\{(v_1, \dots, v_K) \in [lb, 1]^K \;\big|\; \max\{v_1, \dots, v_K\} = 1\big\} \tag{17}$$

of predictor weights for a given lower bound lb of (about) 10^{-8}. This restriction can either be incorporated explicitly with a corresponding definition of the search space for the outer optimizer or implicitly by transforming elements of a (potentially much) bigger search space $\widetilde{\mathcal{V}}$ into elements of $\mathcal{V}$.

With the latter method it is very easy to use the broad class of box constrained optimizers, because the mapping (12) obviously transforms elements of $\widetilde{\mathcal{V}} = [lb, 1]^K$ into elements of $\mathcal{V}$. Since it is just as easy to construct mappings of $\widetilde{\mathcal{V}} = \mathbb{R}^K$ into $\mathcal{V}$, even unconstrained optimizers may be employed if desired. However, these many-to-one mappings of the outer optimizer's search space into the set of input values for the objective function clearly lead to many-to-one mappings between elements of the search space and values of the objective function, which may handycap the outer optimizer considerably.

Explicitly using $\mathcal{V}$ as the outer optimizer's search space however has the disadvantage that the constraint $\max\{v_1, \dots, v_K\} = 1$ is not a simple box constraint, which at first prohibits the usage of many optimizers only capable of managing box constraints. To remedy this problem, we propose to split the outer optimization problem into K subproblems which have a reduced problem dimension of $K - 1$ and require only simple box constraints. In subproblem k, $k \in \{1,\dots,K\}$, the input variable v_k is fixed to 1, whereas the search space (for the remaining $K-1$ input variables) is set to $\widetilde{\mathcal{V}}_k := [lb, 1]^{K-1}$. More precisely, in the k-th subproblem, elements of the search space $\widetilde{\mathcal{V}}_k$ are mapped to elements of

$$\mathcal{V}_k := \left\{(v_1,\dots,v_K) \in [lb,1]^K \,\middle|\, v_k = 1\right\} \tag{18}$$

via the mapping

$$\widetilde{\mathcal{V}}_k \to \mathcal{V}_k; \quad (\widetilde{v}_1,\dots,\widetilde{v}_{K-1}) \mapsto \big(\underbrace{\widetilde{v}_1,\dots,\widetilde{v}_{k-1}}_{v_1,\dots,v_{k-1}}, \underbrace{1}_{v_k}, \underbrace{\widetilde{v}_k,\dots,\widetilde{v}_{K-1}}_{v_{k+1},\dots,v_K}\big). \tag{19}$$

Obviously, $\mathcal{V}_k \subset \mathcal{V}$ because of $\max\{v_1,\dots,v_K\} = v_k = 1$ for each $(v_1,\dots,v_K) \in \mathcal{V}_k$, and $\cup_{i=1}^K \mathcal{V}_k = \mathcal{V}$, so all feasible predictor weights are reachable by the outer optimization.[24] The disadvantage of our proposal is the additional computational burden of solving K optimization problems of dimension $K-1$ instead of solving only one problem of dimension K, but we experienced that getting rid of the many-to-one transformation is well worth the effort.

As a last, but very beneficial refinement for the outer optimizer's search space and its transformation, we propose to use a logarithmic representation of $\widetilde{\mathcal{V}}_k$ in the k-th subproblem, in order to help the outer optimizer to reach values near the lower bound lb much more easily. More precisely, this finally leads, for $k = 1,\dots,K$, to $\widetilde{\mathcal{V}}_k := [\log_{10}(lb), 0]^{K-1}$ as the search space of dimension $K-1$ and to the mapping

$$\widetilde{\mathcal{V}}_k \to \mathcal{V}_k; \quad (\widetilde{v}_1,\dots,\widetilde{v}_{K-1}) \mapsto \big(\underbrace{10^{\widetilde{v}_1},\dots,10^{\widetilde{v}_{k-1}}}_{v_1,\dots,v_{k-1}}, \underbrace{1}_{v_k}, \underbrace{10^{\widetilde{v}_k},\dots,10^{\widetilde{v}_{K-1}}}_{v_{k+1},\dots,v_K}\big) \tag{20}$$

as the one-to-one transformation of the search space into the set of feasible predictor weights $\mathcal{V} = \cup_{i=1}^K \mathcal{V}_k$.

With the proposed definition of the outer optimizer's search space, the problem at hand calls for a constrained global optimizer which is capable of managing simple box constraints. Mullen (2014) presents a very detailed survey of continuous global optimizers in R as well as a thorough empirical comparison of the corresponding implementations. Our reference implementation, the R package **MSCMT**[25], features a modular design which allows to plug in many different optimizers for the outer optimization, the current package version includes support for most of the optimizers mentioned in Mullen (2014) as well as some further optimizers.

Below, the performance of 16 different outer optimizers will be compared. One of the optimizers in this survey is a newly introduced implementation of (Gilli et al., 2011, Al-

[24]Of course, the sets $\mathcal{V}_k$ are not pairwise disjoint, but this does not raise trouble for the optimizer in the particular subproblems.

[25]See Becker and Klößner (2017b).

gorithm 49), which we call `DEoptC`. Compared to the already available R-implementation in function `DEopt` of R package **NMOF**, `DEoptC` combines a Differential Evolution algorithm with the proposed parametrization of the search space, a stopping rule based on lack of improvement for a given number of generations and memory pre-allocations for the Fortran-calls to `WNNLS` in a pure C-implementation, resulting in a speed multiplier of more than four.

3.6 An Algorithm for Solving Generalized SCM Tasks

Overall, the ingredients described in the previous subsections are put together to form our algorithm for solving generalized SCM tasks, see Figure 1.

```
   Input: matrices $\widetilde{X}$, $\widetilde{Z}$, lower bound $lb$, mapping $V = V(v_1, \ldots, v_K) = V(v)$
   Output: optimal $v^* = (v_1^*, \ldots, v_K^*)$ and $w^* = (w_1^*, \ldots, w_J^*)$
1  $\mathcal{J} \leftarrow$ columns corresponding to sunny donors
2  if $\mathcal{J}$ is empty then
3      $w^* \leftarrow \arg\min_w w'\widetilde{Z}'\widetilde{Z}w$ s.t. $\widetilde{X}w = 0, w \geq 0, \mathbb{1}'w = 1$
4      $v^* \leftarrow$ equal weights (or other arbitrary choice)
5  else
6      if $\mathcal{J}$ is a singleton then
7          set $w^*$ accordingly
8          $v^* \leftarrow$ equal weights (or other arbitrary choice)
9      else
10         if unrestricted outer optimum is feasible then
11             $w^* \leftarrow \arg\min_w w'\widetilde{Z}'\widetilde{Z}w$ s.t. $w \geq 0, \mathbb{1}'w = 1$
12             $v^* \leftarrow$ a particular $v$ with $w^* = W^*(v) = \arg\min_w w'\widetilde{X}'V(v)\widetilde{X}w$
13         else
14             remove columns of $\widetilde{X}$ and $\widetilde{Z}$ not contained in $\mathcal{J}$
15             $v^* \leftarrow \arg\min_v W^*(v)'\widetilde{Z}'\widetilde{Z}W^*(v)$ s.t. $lb \leq v_k \leq 1$, $\max v_k = 1$
16             $w^* \leftarrow W^*(v^*)$
17             re-impute 0's to $w^*$ for components belonging to removed donors
```

Figure 2: Algorithm for Solving Generalized SCM Tasks

As a first step, the subset of donors that are sunny has to be determined using a standard solver for linear programs[26]. If there are no sunny donors, i.e., if the inner optimization can – independent of the predictor weights – be solved perfectly, the quadratic program given by equation (10) has to be solved (preferably using the `WNNLS` algorithm)

[26]In our R package **MSCMT**, we use the R package **lpSolve** for solving linear programs, see Berkelaar and others (2015).

to find optimal (with respect to the outer objective function) donor weights among those with a perfect fit with respect to the inner optimization.

In a second step, the (rarely occuring) special case of only one sunny donor can be sorted out, because there is obviously no need for an optimization in this case: all weight will be put on the unique sunny donor, independent of the predictor weights $v_1, \dots, v_K$.

In the next step, the unrestricted outer optimum has to be calculated using equation (13) (preferably using the `WNNLS` algorithm) and checked for feasibility using equation (16) (and a standard solver for linear programs) in order to avoid solving the difficult and typically time-consuming outer optimization task when indicated.

In case of a feasible unrestricted outer optimum, the algorithm terminates with the unrestricted outer optimum, while in the remaining case the nested optimization task has actually to be solved. To potentially improve numerical stability and reduce computational burden, the data belonging to shady donors can be removed first, because we know from the theory developed above that shady donors will always obtain zero weights in $W^*(v_1, \dots, v_K)$, regardless of the values the predictor weights take.

4 Reliability of SCM Implementations

In the following, we investigate the reliability of available implementations for synthetic control methods. More precisely, we compare the results obtained with our implementation of the newly proposed algorithm, the R package **MSCMT**[27] (R/**MSCMT**), with other implementations of synthetic control methods, namely the R package **Synth** version 1.1-5 (R/**Synth**), Matlab version R2016b together with the Synth code from Jens Hainmueller's webpage, and Stata (version 14) in combination with the corresponding Synth package by Alberto Abadie, Alexis Diamond, and Jens Hainmueller (version 0.0.7)[28].

A terse but similar comparison has already been provided by Becker and Klößner (2017a), where R/**MSCMT** is shown to be the only implementation finding the correct (optimal) solution.[29] In the sequel, we will illustrate that the application in Becker and Klößner (2017a) is not a sole exception, by replicating some of the estimations of Abadie and Gardeazabal (2003) with the implementations mentioned above. More precisely, our investigation will cover two applications: the 'main' application with the Basque Country as treated unit, and the 'placebo' result for Catalonia (Cataluna) as treated unit.[30] Finally, we will compare the performance of different numerical optimizers when

[27]We use R version 3.3.2 and **MSCMT** version 1.2.0, see R Core Team (2016) and Becker and Klößner (2017b).

[28]See Abadie et al. (2011) and `http://web.stanford.edu/~jhain/synthpage.html`.

[29]This application also provides a practical example where the unrestricted outer optimum is actually feasible.

[30]Reproducible calculations of the results obtained with packages **MSCMT** and **Synth** are contained in the web appendix. All analyses are based on the dataset `basque` and the data preparation described on the help page for function `synth`, both provided by R package **Synth**.

used for solving the outer optimization, separately for the Basque Country and Catalonia as treated unit.

4.1 Synthetic Control Unit for the Basque Country

The compositions of the Basque Country's synthetic control unit and the RMSPEs of the dependent variable obtained with the aforementioned implementations are collected in Table 2.

	Matlab	R/**MSCMT**	R/**Synth**	Stata
Baleares (Islas)	0.00000	21.92728	0.00000	0.00000
Cataluna	85.19074	63.27857	85.08140	90.51304
Madrid (Comunidad De)	14.80926	14.79414	14.91860	9.48685
RMSPE GDP per Capita 1960–69	0.09416	0.06547	0.09415	0.10484

Table 2: Results for the Basque Country. Weights of donor units in % and RMSPE of GDP per Capita 1960–69. Donor units with (nearly) zero weights are omitted.

The results obtained with R/**Synth**[31] coincide with the results reported in Abadie and Gardeazabal (2003). Obviously, Matlab, R/**Synth** and Stata[32] deliver suboptimal (and thus wrong) results, as the RMSPE obtained with R/**MSCMT**[33] is considerably smaller than the RMSPEs obtained with the other implementations.

As Figure 3 shows, the synthetic control unit obtained with R/**MSCMT** differs considerably from the other results not only with respect to the donor weights and the outer objective function value, but also with respect to the predicted values for the real per capita GDP. All in all, the differences between the values for the (true) Basque Country and its synthetic control unit are considerably increased for the improved (in terms of the objective function value) synthetic control unit obtained with R/**MSCMT**.

Table 3 summarizes the 'optimal' predictor weights found by the outer optimizer for the different implementations.

[31]We performed the calculation with all four possible combinations of inner (`ipop` vs. `LowRankQP`) and outer (default vs. `genoud`) optimizers and report only the best result, which is obtained by using `LowRankQP` in combination with `genoud`.

[32]We use the most extensive optimization method by enabling options `nested` and `allopt`.

[33]All parameters were left at their defaults, the random seed has been set to 1.

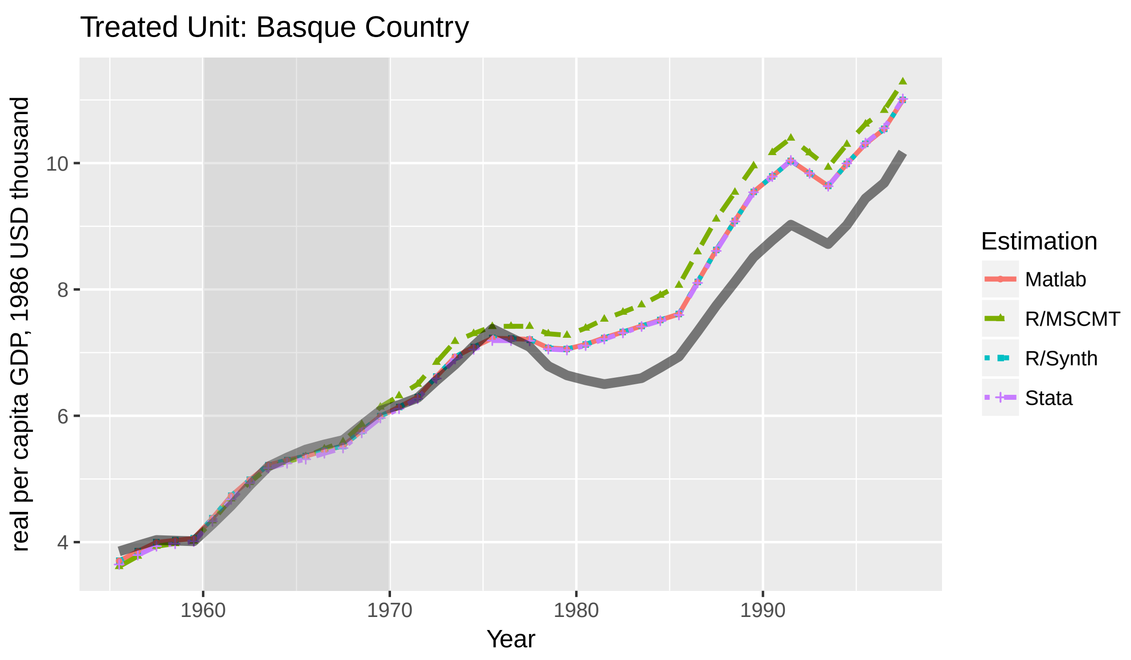

Figure 3: Comparison of the synthesized real per capita GDP for all synthetic control units and the true real per capita GDP (thick line) for the Basque Country. The shaded area depicts the optimization period.

	Matlab	R/**MSCMT**	R/**Synth**	Stata
school.illit	0.00978	0.00158	1.29933	6.06099
school.prim	2.78902	0.00158	0.38459	8.97528
school.med	9.87364	0.00158	14.88152	6.79965
school.higher	10.64217	0.02903	13.98743	1.41176
invest	0.05378	0.02990	0.36511	0.75721
gdpcap	10.52137	99.92528	1.87268	22.92560
sec.agriculture	8.07297	0.00158	10.33610	5.61804
sec.energy	8.66569	0.00158	7.84373	9.42606
sec.industry	8.96004	0.00158	11.18351	1.37489
sec.construction	2.43919	0.00158	5.91204	2.23459
sec.services.venta	8.57422	0.00158	0.42320	0.04532
sec.services.nonventa	4.96361	0.00158	9.21059	15.74805
popdens	24.43452	0.00158	22.30016	18.62256

Table 3: Results for the Basque Country. Optimal predictor weights in % (as reported by software).

To illustrate that the main culprit of the suboptimal solutions obtained with Matlab, R/**Synth** and Stata is the outer optimizer and, potentially, the parametrization of the search space, we plug the optimal predictor weights reported by R/**MSCMT** as fixed predictor weights into the inner optimizers of Matlab, R/**Synth** (`ipop` as well as `LowRankQP`) and Stata, bypassing the outer optimization. Table 4 reports the results of this experiment: all alternative implementations now report considerably improved outer objective function values (RMSPEs), clearly indicating that the outer optimizers originally failed to find (near) optimal predictor weights. While the predictor losses for LowRankQP, Matlab and Stata are (nearly) identical to the predictor loss obtained with R/**MSCMT**, ipop's result is clearly distinguishable from R/**MSCMT**'s result, indicating additional problems with ipop as inner optimizer.[34]

	ipop	LowRankQP	Matlab	Stata	WNNLS
Andalucia	0.02890	0.00000	0.00054	0.00000	0.00000
Aragon	0.05736	0.00000	0.00218	0.00000	0.00000
Principado De Asturias	0.07121	0.00000	0.00335	0.00000	0.00000
Baleares (Islas)	24.84337	21.96958	22.79723	21.93067	21.92728
Canarias	0.07195	0.00000	0.00230	0.00000	0.00000
Cantabria	0.51458	0.02273	0.39857	0.00182	0.00000
Castilla Y Leon	0.03800	0.00000	0.00094	0.00000	0.00000
Castilla-La Mancha	0.02198	0.00000	0.00037	0.00000	0.00000
Cataluna	57.56800	63.18231	61.41685	63.27086	63.27857
Comunidad Valenciana	0.06862	0.00000	0.00325	0.00000	0.00000
Extremadura	0.02594	0.00000	0.00032	0.00000	0.00000
Galicia	0.03258	0.00000	0.00067	0.00000	0.00000
Madrid (Comunidad De)	16.53677	14.82538	15.36867	14.79664	14.79414
Murcia (Region de)	0.03725	0.00000	0.00090	0.00000	0.00000
Navarra (Comunidad Foral De)	0.05410	0.00000	0.00259	0.00000	0.00000
Rioja (La)	0.02939	0.00000	0.00127	0.00000	0.00000
RMSPE GDP per Capita 1960–69	0.06615	0.06548	0.06566	0.06547	0.06547
Predictor loss ($\times 10\,000$)	3.49127	3.37496	3.37939	3.37496	3.37496

Table 4: Results for the Basque Country obtained with fixed, optimal predictor weights as reported by R/**MSCMT**. Weights of donor units in % and RMSPE of GDP per Capita 1960–69. Donor units with (nearly) zero weights are omitted.

The results of Table 4 confirm our conjecture that using a local outer optimizer with only one or a few different starting values, a strategy incorporated by Matlab, R/**Synth** (when using the default outer optimizer) and Stata, may be an inappropriate approach. Since R/**Synth** in combination with genoud, a genetic global optimizer, also fails to obtain a (near) optimal solution, the parametrization of the search space seems to have considerable influence on the success of the outer optimization as well.

4.2 Synthetic Control Unit for Catalonia

The compositions of Catalonia's synthetic control unit and the RMSPEs of the dependent variable obtained with the different implementations of synthetic control methods are collected in Table 5. Again, Matlab, R/**Synth** and Stata deliver suboptimal results, as the RMSPE of R/**MSCMT**'s result is considerably lower.

[34]See the web appendix for more serious examples of ipop's suboptimality.

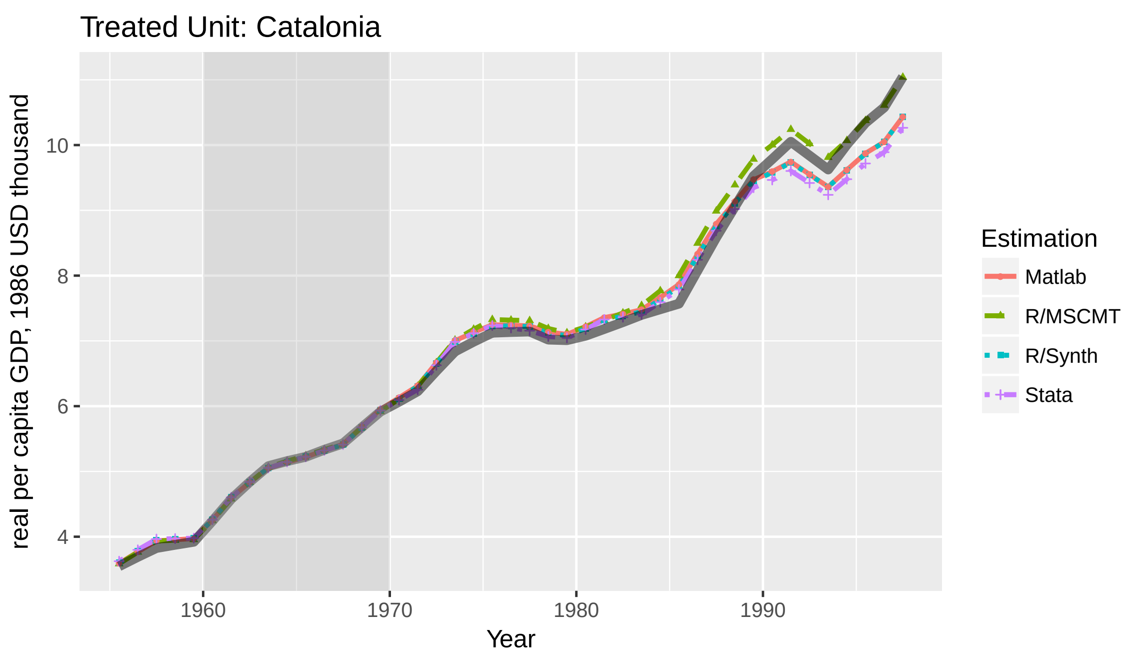

Figure 4: Comparison of the synthesized real per capita GDP for all synthetic control units and the true real per capita GDP (thick line) for Catalonia. The shaded area depicts the optimization period.

	Matlab	R/**MSCMT**	R/**Synth**	Stata
Principado De Asturias	6.28543	0.00000	1.94288	21.46473
Baleares (Islas)	28.74754	23.24732	27.41967	25.62794
Cantabria	22.35530	0.00000	24.04876	7.31928
Madrid (Comunidad De)	42.61172	43.78377	44.27370	45.58805
Murcia (Region de)	0.00000	0.00000	2.31422	0.00000
Navarra (Comunidad Foral De)	0.00000	32.96891	0.00002	0.00000
RMSPE GDP per Capita 1960–69	0.01917	0.00897	0.01741	0.01719

Table 5: Results for Catalonia. Weights of donor units in % and RMSPE of GDP per Capita 1960–69. Donor units with (nearly) zero weights are omitted.

Although neither the donor weights nor the outer objective function value are reported in Abadie and Gardeazabal (2003), a comparison of our Figure 4 with Figure 4 of Abadie and Gardeazabal (2003) indicates that the synthesized real per capita GDP values obtained with Matlab and R/**Synth**[35] essentially coincide with the results of Abadie and Gardeazabal (2003), while R/**MSCMT**[36] and Stata[37] clearly produce distinctly different compositions of the synthetic control unit as well as synthesized real per capita GDP values.

[35]As above, we performed the calculation with all four possible combinations of inner (`ipop` vs. `LowRankQP`) and outer (default vs. `genoud`) optimizers and report only the best result, which this time is obtained by using `ipop` in combination with `genoud`.

[36]As above, all parameters were left at their defaults, the random seed has been set to 1.

[37]As above, we use the most extensive optimization method by enabling options `nested` and `allopt`.

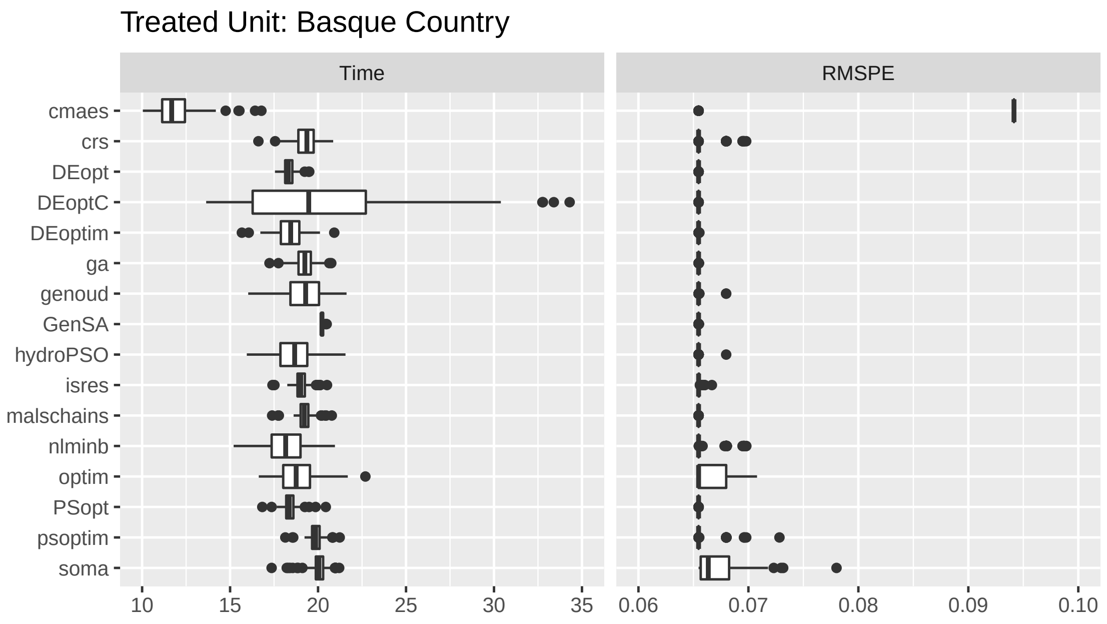

Figure 5: Comparison of box plots for time (in seconds) and RMSPE (outer objective function value) for various optimizers (based on 250 random seeds) with Basque country as treated unit.

4.3 Comparison of Outer Optimizers

To illustrate that calculating synthetic controls potentially is a very demanding task for the outer optimizer, we compare the results of 16 different optimizers. Apart from 13 optimizers surveyed in Mullen (2014),[38] we include `DEoptC` (introduced in Subsection 3.5 above) and two deterministic box-constrained local optimizers, `nlminb` and `optim` (with method `"L-BFGS-B"`) from R package **stats**. The latter two are made stochastic and tuned for global optimization tasks by incorporating repeated optimizations with a given number of randomly generated starting values.

To compare the performance ot the optimizers, we repeated the calculation of the Basque country's synthetic control unit 250 times, varying the initial seed of the random number generator(s) from 1 to 250. To establish a fair comparison between the optimizers, we tried hard to adjust the mean time spent for single calculations to about 20 seconds[39], while maintaining sensible and comparable parameter settings.

The results for Basque country are depicted in Figure 5. Obviously, most outer optimizers found a (near) optimal solution in the vast majority of runs. `optim` was successful at least in the majority of runs, while `soma` mostly and `cmaes` nearly always failed. But notice that not a single result is worse than the solutions obtained with Matlab, R/**Synth**,

[38]Some optimizers from Mullen (2014) were excluded because they are not suited well for the particular class of optimization problems involved with SCM.

[39]We used a single core of an Intel® i7 860@2.80GHz. We failed to find appropriate settings to adjust the time for `cmaes` to about 20 seconds. The non-default parameters for the various optimizers are listed in the web appendix.

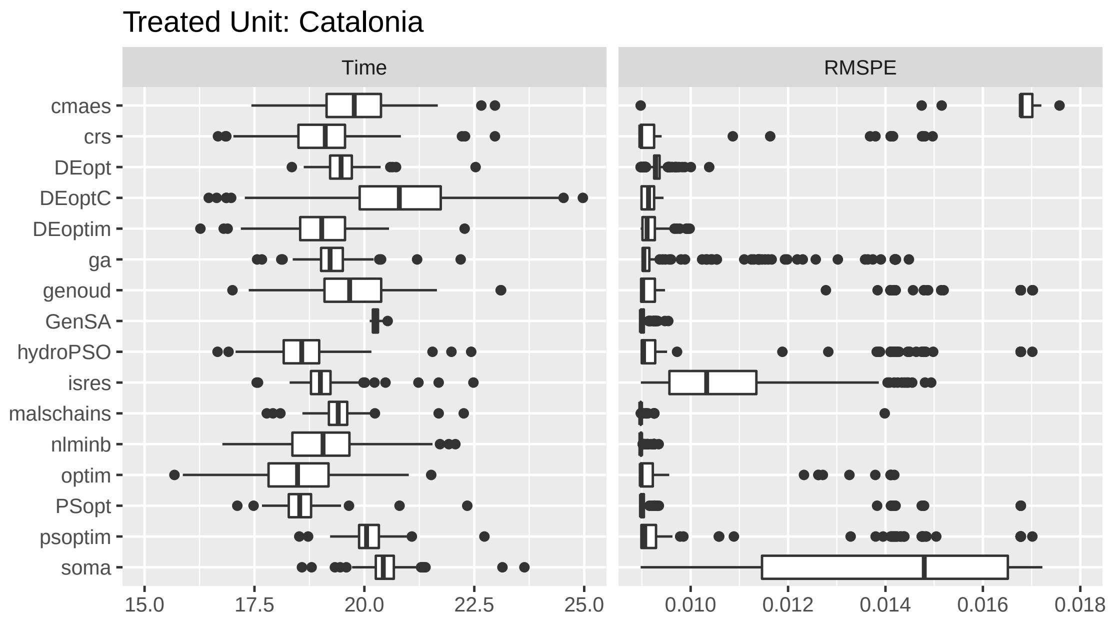

Figure 6: Comparison of box plots for time (in seconds) and RMSPE (outer objective function value) for various optimizers (based on 250 random seeds) with Catalonia as treated unit.

or Stata, which have RMSPEs of at least 0.09415.

Repeating the analysis for the calculation of Catalonia's synthetic control unit reveals the difficulties which may arise in the outer optimization, see Figure 6. Here, many optimizers often fail to obtain a (near) optimal solution. In fact, one has to be lucky to find a (near) optimal solution in the first attempt, and it certainly is advisable to repeat the calculation many times (with different seeds) and/or using various outer optimizers.

As above, all results obtained in this study are at least as good as the results obtained with R/**Synth**, Matlab, or Stata, the only exception being some outliers for `cmaes`. Our implementation `DEoptC` features a good overall performance and has a very small worst-case loss.

5 Conclusion

In this paper, we contribute to the literature on synthetic control methods by developing helpful results on the theory of the optimization problems underlying synthetic control methods. In particular, we provide a unifying framework for general SCM tasks which incorporates both the 'standard' SCM approach of Abadie and Gardeazabal (2003) and the MSCMT approach of Klößner and Pfeifer (2015). Furthermore, we show that predictor weights must be bounded away from zero in order to obtain a mathematically sound optimization problem with an objective function that is continuous on its compact domain of definition.

As existing implementations of SCM have recently be shown to be rather unreliable, we develop an algorithm for solving such general SCM tasks. During its first stages, this algorithm checks for several special cases, thereby improving speed and accuracy of calculations in those cases. For instance, if a perfect fit is possible with respect to the economic predictors, only a nonnegatively constrained linear least squares problem with linear equality constraints has to be solved to optimize the fit with respect to the outcome, which can be done fast and numerically stable by using the `WNNLS` algorithm. Additionally, if the unrestricted outer optimum is feasible, i.e., if predictor weights can be found for which the donor weights that optimize the fit with respect to the outcome constitute the solution to the inner optimization problem, then there is no need to solve a nested optimization problem: in this case, optimal donor weights are given by the feasible unrestricted outer optimum and can be computed fast and numerically stable by using the `WNNLS` algorithm, again.

We also elaborate on cleverly searching for synthetic controls when iterative optimizers have to be used for calculations, with special emphasis on reliability of results and computation speed. On the one hand, by removing so-called 'shady' donors, speed of computation and numerical accuracy may be fostered. On the other hand, using `WNNLS` for solving the inner optimization improves significantly upon existing implementations with respect to accuracy, and even more so with respect to speed of computation. Considering the outer optimization task, one should avoid opting for local optimizers that use only one or a few starting values, as such implementations of SCM often fail to find the correct solution. A comparison of various numerical optimizers one may use for solving the outer optimization shows that stochastic, heuristic methods are much more successful in finding the correct solution. However, as some instances of SCM calculations are very demanding for the outer optimization task, it is advisable to rerun calculations using different initial seeds of the random number generator and/or different heuristic optimizers to increase the probability of finding the correct optimum.

With the R package **MSCMT**, we provide an open source implementation of our new algorithm for calculating synthetic control units for a free open source software application. With its modular design, **MSCMT** features interfaces to various optimizers, including `WNNLS` for the inner optimization task and many different heuristic methods such as simulated annealing, genetic algorithms, differential evolution, and particle swarm optimizers for the outer optimization. All methods described in this paper are thus immediately applicable by applied researchers.

References

Abadie, Alberto, Alexis Diamond, and Jens Hainmueller, "Synthetic Control Methods for Comparative Case Studies: Estimating the Effect of California's Tobacco Control Program," *Journal of the American Statistical Association*, 2010, *105* (490), 493–505.

_ , _ , and _ , "Synth: An R Package for Synthetic Control Methods in Comparative Case Studies," *Journal of Statistical Software*, 6 2011, *42* (13), 1–17.

_ , _ , and _ , "Comparative Politics and the Synthetic Control Method," *American Journal of Political Science*, 2015, *59* (2), 495–510.

_ and Javier Gardeazabal, "The Economic Costs of Conflict: A Case Study of the Basque Country," *The American Economic Review*, 2003, *93* (1), 113–132.

Acemoglu, Daron, Simon Johnson, Amir Kermani, James Kwak, and Todd Mitton, "The value of connections in turbulent times: Evidence from the United States," *Journal of Financial Economics*, 2016, *121* (2), 368 – 391.

Becker, Martin and Stefan Klößner, "Estimating the Economic Costs of Organized Crime By Synthetic Control Methods," 2017. Working Paper.

_ and _ , *MSCMT: Multivariate Synthetic Control Method Using Time Series* 2017. R package version 1.2.0.

Berkelaar, Michel and others, *lpSolve: Interface to 'Lp_solve' v. 5.5 to Solve Linear/Integer Programs* 2015. R package version 5.6.13.

Cavallo, Eduardo, Sebastian Galiani, Ilan Noy, and Juan Pantano, "Catastrophic Natural Disasters and Economic Growth," *The Review of Economics and Statistics*, 2013, *95* (5), 1549–1561.

Gardeazabal, Javier and Ainhoa Vega-Bayo, "An Empirical Comparison Between the Synthetic Control Method and Hsiao et al.'s Panel Data Approach to Program Evaluation," *Journal of Applied Econometrics*, 2016, pp. n/a–n/a.

Gilli, Manfred, Dietmar Maringer, and Enrico Schumann, *Numerical Methods and Optimization in Finance*, Waltham, MA, USA: Academic Press, 2011. ISBN 0123756626.

Gobillon, Laurent and Thierry Magnac, "Regional Policy Evaluation: Interactive Fixed Effects and Synthetic Controls," *The Review of Economics and Statistics*, 2016, *98* (3), 535–551.

Hanson, Richard J. and Karen H. Haskell, "Algorithm 587: Two Algorithms for the Linearly Constrained Least Squares Problem," *ACM Trans. Math. Softw.*, September 1982, *8* (3), 323–333.

Haskell, Karen H. and Richard J. Hanson, "An algorithm for linear least squares problems with equality and nonnegativity constraints," *Mathematical Programming*, 1981, *21* (1), 98–118.

Hsiao, Cheng, H. Steve Ching, and Shui Ki Wan, "A Panel Data Approach for Program Evaluation: Measuring the Benefits of Political and Economic Integration of Hong Kong with Mainland China," *Journal of Applied Econometrics*, 2012, *27* (5), 705–740.

Jinjarak, Yothin, Ilan Noy, and Huanhuan Zheng, "Capital Controls in Brazil–Stemming a Tide with a Signal?," *Journal of Banking and Finance*, 2013, *37* (8), 2938–2952.

Karatzoglou, Alexandros, Alex Smola, Kurt Hornik, and Achim Zeileis, "kernlab – An S4 Package for Kernel Methods in R," *Journal of Statistical Software*, 2004, *11* (9), 1–20.

Kaul, Ashok, Stefan Klößner, Gregor Pfeifer, and Manuel Schieler, "Synthetic Control Methods: Never Use All Pre-Intervention Outcomes as Economic Predictors," October 2016. Working Paper.

Kleven, Henrik Jacobsen, Camille Landais, and Emmanuel Saez, "Taxation and International Migration of Superstars: Evidence from the European Football Market," *The American Economic Review*, 2013, *103* (5), 1892–1924.

Klößner, Stefan and Gregor Pfeifer, "Synthesizing Cash for Clunkers: Stabilizing the Car Market, Hurting the Environment," Annual Conference 2015 (Muenster): Economic Development - Theory and Policy, Verein für Socialpolitik / German Economic Association 2015.

Mullen, Katharine, "Continuous Global Optimization in R," *Journal of Statistical Software*, 2014, *60* (1), 1–45.

Ormerod, John T. and M. P. Wand, *LowRankQP: Low Rank Quadratic Programming* 2014. R package version 1.0.2.

Pinotti, Paolo, "The Economic Costs of Organised Crime: Evidence from Southern Italy," *The Economic Journal*, 2015, *125* (586), F203–F232.

R Core Team, *R: A Language and Environment for Statistical Computing* R Foundation for Statistical Computing 2016.

Soetaert, K., K. van den Meersche, and D. van Oevelen, *limSolve: Solving Linear Inverse Models* 2009. R package version 1.5.1.

Wolfe, Philip, "The Simplex Method for Quadratic Programming," *Econometrica*, 1959, *27* (3), 382–398.