# Computing Generalized Synthetic Controls with the R package MSCMT

Martin Becker
Statistics and Econometrics
Saarland University

Stefan Klößner*
Statistics and Econometrics
Saarland University

Preliminary version, work in progress: July 27, 2016

**Abstract**

In this paper, we develop new methods for calculating synthetic control units. In particular, we improve on existing implementations both in terms of accuracy of the provided solutions as well as speed of computation. Furthermore, we show how to detect and treat important special cases that have not been adressed in the literature until now. Our new methods are availabe in R package MSCMT (Multivariate Synthetic Control Methods using Time series) and can be applied to generalizations of 'standard' synthetic control methods, too.

*Correspondence to: Stefan Klößner, Saarland University, Campus C3 1, 66123 Saarbrücken, Germany. E-mail: S.Kloessner@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. However, there is only one publicly available implementation of SCM methods that can be used free of charge, the R package Synth.[1] Unfortunately, the routines supplied by R package Synth often produce unreliable or plainly wrong results, furthermore, running these routines usually takes quite a lot of time. To the best of our knowledge, the same holds true for the implementations of Synth that are available for commercial software packages Stata and Matlab.

Therefore, in this paper, we provide helpful results on the theory of the optimization problems that have to be solved when a synthetic control is calculated. The benefit of these results is, on the one hand, that the dimension of the problem may be significantly reduced, helping to speed up calculations. On the other hand, the theoretical results enable us to detect important special cases that have been overlooked by the literature. To these special cases, we provide algorithmic solutions for detecting and solving them, resulting in exact results for the synthetic control which can be calculated extremely fast.

We also elaborate on cleverly attacking the time-consuming search for the synthetic control when iterative optimizers have to be used for calculations, with special emphasis on numerical stability and computation speed. All these routines, have been implemented in R package MSCMT, which is publicly available.[2] Additionally, MSCMT provides a plethora of helpful features for SCM applications, e.g. fast calculation of placebo studies by using parallel computing, handy plots, p values, and so on. As its name (**M**ultivariate **S**ynthetic **C**ontrol **M**ethod using **T**ime Series) suggests, package MSCMT can handle not only 'standard' SCM applications, but also the generalized MSCMT approach of Klößner and Pfeifer (2016) which allows to consider several variables of interest simultaneously and to treat explanatory 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 (2016), 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. Section 4 illustrates R package MSCMT, while section 5 concludes. Some theory and proofs have been relegated to an appendix.

# 2 The Synthetic Control Method

## 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

[1] See R Core Team (2016), Abadie et al. (2011). There also exist implementations by the same authors for Stata and Matlab.

[2] See Becker and Klößner (2016).

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 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 $T$ 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 and the counterfactual becomes as small as possible. The distance metric used to measure this difference is: $\|X_1 - X_0W\|_V = \sqrt{(X_1 - X_0W)' V (X_1 - X_0W)}$, 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 control unit 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 subset of $Y$ for the treated unit over the pre-intervention periods, while $Z_0$ denotes the analogous 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 (2016), 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 (2016) extend the equations

[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).

(1) and (2) for the approximation errors with respect to predictor and outcome data to the following ones:[5]

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

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

To explain the formulas (3) and (4), we adopt the notation of Klößner and Pfeifer (2016) and denote the outcome $Y$ of the dependent variable $l$ at time $m$ for unit $j$ by $Y_{l,m,j}$, with $l = 1,\dots,L$ running over $L$ variables of interest and $m = 1,\dots,M_l^{\text{pre}}$ running over the $M_l^{\text{pre}}$ pre-treatment observations of variable $l$. As above, $j = 1,\dots,J+1$ runs through the units, with $j = 1$ denoting the treated unit and $j = 2,\dots,J+1$ for the control units. The values of $K$ economic predictors are denoted similarly by $X_{k,n,j}$, with $k = 1,\dots,K$ running over all economic predictors and $n = 1,\dots,N_k$ running over the pre-treatment observation times of economic predictor $k$.

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

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

allowing for

- a number $L$ of possibly more than two dependent variables,
- weights $\alpha_1,\dots,\alpha_L$ 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 $\beta_{l,1},\dots,\beta_{l,M_l^{\text{pre}}}$ for the pre-treatment values of $l$-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

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

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

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

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

[6]Obviously, instead of minimizing $\Delta_X(v_1,\dots,v_K,W)$, the root mean square error of the predictors' approximation as given in (6), one can equivalently minimize $\Delta_X^2(v_1,\dots,v_K,W)$, 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 $\widetilde{Z} := Z_0 - Z_1\iota'$,[7] $\widetilde{X} := X_0 - X_1\iota'$, and

$$V(v_1, \ldots, v_K) := \begin{pmatrix} v_1 & 0 & \ldots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \ldots & 0 & v_K \end{pmatrix},$$

the standard SCM approach can be described as the following two-step procedure:

- 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 setting

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

with $I_{N_k}$ denoting the $N_k$-dimensional identity 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] $\iota$ denotes the vector of ones.

and

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

In the sequel, we will therefore, both with respect to theory and algorithms, discuss the following general SCM problem:

- given a linear mapping $V$ which maps $v_1, \ldots, v_K$ to a diagonal matrix with non-negative entries,
- a matrix $\widetilde{X}$, used in the inner optimization to get $W^*(v_1, \ldots, v_K)$ as the minimizer of $w'\widetilde{X}'V(v_1, \ldots, v_K)\widetilde{X}w$ over all non-negative $w$ whose components sum to unity,
- and a matrix $\widetilde{Z}$, used in the outer optimization:
- determine $v_1^*, \ldots, v_K^*$ such that $W^*(v_1, \ldots, v_K)\widetilde{Z}'\widetilde{Z}W^*(v_1, \ldots, v_K)$ becomes as small as possible.

# 3 Solving Generalized SCM Problems

## 3.1 Dimension Reduction & A Special Case

In order to speed up the inner optimization as well as to detect an important special case, 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 these squared differences, 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 $\widetilde{X}$ somehow 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 $H := \mathrm{conv}(\{\widetilde{x}_1, \ldots, \widetilde{x}_J\})$ the convex hull of the columns of $\widetilde{X}$, we call $\widetilde{x}_j$ ($j = 1, \ldots, J$) **sunny** (w.r.t. $H$ or w.r.t. $\widetilde{x}_1, \ldots, \widetilde{x}_J$) 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.[8] In Proposition 1 in the appendix, we show that if $w^*$ is a minimizer of the inner objective function with $(w^*)'\widetilde{X}'V(v_1, \ldots, v_K)\widetilde{X}w^* > 0$,

[8] A mathematically rigorous treatment can be found in the appendix, starting at Definition 1.

then $w_j^*$ can only be positive for sunny donors. This result is very useful for reducing the dimension of the problem, 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. Furthermore, notice that checking whether a donor is sunny is easily possible, because it amounts to checking the solution of a linear program in $(w_1, \ldots, w_K, \alpha)$, with $\alpha$ denoting the quantity appearing in Proposition 1.

One might wonder whether it is possible that there are no sunny donors at all and what to do if this were to happen. Proposition 1 in the appendix 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 are different vectors of donor weights with such a perfect predictor fit. We therefore treat this special case of 'no sunny donors' by searching among all those vectors of donor weights for one that is optimal with respect to the outer objective function, i.e. for those with the best fit with respect to the outcome data: mathematically, we thus look for non-negative weights $w$ summing to unity which minimize $w'\widetilde{Z}'\widetilde{Z}w$ subject to $\widetilde{X}w = 0$.

## 3.2 Feasibility of the Outer Global Optimum & 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 $w'_{\text{global}}\widetilde{Z}'\widetilde{Z}w_{\text{global}}$, where $w_{\text{global}}$ denotes a minimizer of $w'\widetilde{Z}'\widetilde{Z}w$ over all non-negative $w$ whose components sum to unity, the so-called global optimum. Ideally, we would like optimal predictor weights $v_1^*, \ldots, v_K^*$ to be such that $W^*(v_1^*, \ldots, v_K^*) = w_{\text{global}}$, a case for which the inner and outer objective function are not at odds with each other: in this case, we call the global optimum $w_{\text{global}}$ feasible with respect to the inner minimization. As computing $w_{\text{global}}$ is an easy task because it is a quite standard quadratic optimization problem, it would be very helpful if there was also an easy way of checking whether $w_{\text{global}}$ is feasible.

Applying Proposition 2 given in the appendix to $B := \widetilde{X}'V(v_1, \ldots, v_K)\widetilde{X}$ shows that $w$ is, for given $v_1, \ldots, v_K$, a minimizer of the inner objective function if and only if all components of $\widetilde{X}'V(v_1, \ldots, v_K)\widetilde{X}w$ do not fall below $w'\widetilde{X}'V(v_1, \ldots, v_K)\widetilde{X}w$, with equality for all donors $j$ for which the donor weight $w_j$ is positive. Thus, if $w$ is given, for instance $w = w_{\text{global}}$, then we may check whether there exist $v_1, \ldots, v_K$ such that the above conditions are met. 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 $v_1, \ldots, v_K$ such that $w = W^*(v_1, \ldots, v_K)$. Moreover, checking this is an easy task, because $\widetilde{X}'V(v_1, \ldots, v_K)\widetilde{X}w$ as well as $w'\widetilde{X}'V(v_1, \ldots, v_K)\widetilde{X}w$ inherit from $V$ the property that they are linear functions of $v_1, \ldots, v_K$. Therefore, checking whether a given vector of donor weights, $w$, is feasible can be done by solving an appropriate linear program in terms of $v_1, \ldots, v_K$.

Thus, it is possible to check whether the global optimum, $w_{\text{global}}$, is feasible. If so, $w_{\text{global}}$ are optimal weights for synthesizing, and the synthetization task is solved. If not, the inner and outer objective functions constitute conflicting goals, and other, more demanding methods have to be used, to which we will turn later.

## 3.3 The Inner Optimization

When the global optimum is not feasible, there is no easy way to solve the synthetization problem, and an iterative procedure for determining optimal predictor weights $v_1^*, \ldots, v_K^*$ must be used. Because the objective function with respect to $v_1, \ldots, v_K$ is given by $W^*(v_1, \ldots, v_K)'\widetilde{Z}'\widetilde{Z}W^*(v_1, \ldots, v_K)$, it is essential to be able to compute $W^*(v_1, \ldots, v_K)$ both fast and very precisely. Calculating $W^*(v_1, \ldots, v_K)$ must proceed fast, because this quantity is needed for every set of predictor weights that might be considered during the iteration for finding $v_1^*, \ldots, v_K^*$. At the same time, $W^*(v_1, \ldots, v_K)$ must be calculated very precisely, because imprecise approximations might fool the outer optimizer looking for appropriate predictor weights.

Thus, we need a fast and precise algorithm for solving the inner optimization, i.e. finding $W^*(v_1, \ldots, v_K)$ which minimizes $w'\widetilde{X}V(v_1, \ldots, v_K)\widetilde{X}w = ||V(v_1, \ldots, v_K)^{\frac{1}{2}}\widetilde{X}w||^2$ among all non-negative donor weigths summing up to unity. One possibility of speeding up the calculation of $W^*(v_1, \ldots, v_K)$ has already been discussed above: one can reduce the number of columns of $\widetilde{X}$, i.e. the number of donors that must be considered, by keeping only those columns that belong to sunny donors. Depending on the application, this can be a more or less drastical reduction of the problem's dimension. The possibly reduced problem can be solved fast and numerically stable by using the WNNLS algorithm of Hanson and Haskell (1982).

## 3.4 The Outer Optimization

The outer optimization, i.e. the task of minimizing $W^*(v_1, \ldots, v_K)'\widetilde{Z}'\widetilde{Z}W^*(v_1, \ldots, v_K)$, tacitly assumes that $W^*(v_1, \ldots, v_K)$ is uniquely defined as the solution of the inner optimization problem of minimizing $w'\widetilde{X}V(v_1, \ldots, v_K)\widetilde{X}w$. Obviously, however, there is no unique minimizer of the inner optimization when all predictor weights vanish. Moreover, uniqueness of $W^*(v_1, \ldots, v_K)$ regularly fails even when only some of the predictor weights vanish. But $W^*(v_1, \ldots, v_K)$ being theoretically well-defined is not sufficient for being able to calculate this quantity in a numerically stable way: severe numerical problems will occur when the ratio of the smallest predictor weight(s) over the largest predictor weight is extremely small. Although the WNNLS algorithm is designed to work well even for ill-conditioned matrices, it may become unreliable if it must cope with extremely ill-conditioned matrices.[9] Therefore, we strongly recommend to restrict the ratio of $\frac{\min(v_1,\ldots,v_K)}{\max(v_1,\ldots,v_K)}$ not to fall below some lower bound $lb$ which should not be chosen below $10^{-8}$. Such a restriction ensures that calculations are reliable, and in the end, by calculating the maximal order statistics predictor weights, it is possible to check whether the restriction was actually binding or not.

According to the discussion above, we will in the sequel always assume that $v_k \geq lb v_{\widetilde{k}}$ for all $k, \widetilde{k} = 1, \ldots, K$. In particular, this implies that all $v_k$ are positive so that it is possible to reparameterize the outer optimization in terms of $\log v_1, \ldots, \log v_K$, which we

[9]For details on WNNLS, see Haskell and Hanson (1981).

have observed to be advantageous with respect to (speed of) convergence of the approximations supplied for the outer optimization by an iterative optimizer. Additionally, we have also found that is often quite helpful to do $K$ distinct runs of the optimizer, where in each run another component of the predictor weights is fixed to unity (or equivalently, the corresponding log is fixed to zero), while all other components vary between $lb$ and 1 (or equivalently, their logs vary between $\log(lb)$ and 0).

## 3.5 Algorithm for Solving Generalized SCM Problems

Overall, the ingredients described in the previous subsections are put together to form the algorithm given in Figure 1 for solving a generalized SCM problem.

**Input:** matrices $\widetilde{X}$, $\widetilde{Z}$, lower bound $lb$, mapping $V = V(v_1, \ldots, v_K)$
**Output:** optimal $v^* = (v_1^*, \ldots, v_K^*)$ and $w^* = (w_1^*, \ldots, w_J^*)$

$\mathcal{J} \leftarrow$ columns corresponding to sunny donors
**if** $\mathcal{J}$ *is empty* **then**
  $w^* \leftarrow \arg\min_w w'\widetilde{Z}'\widetilde{Z}w$ s.t. $\widetilde{X}w = 0, w \geq 0, \iota' w = 1$
  $v^* \leftarrow$ equal weights (or other arbitrary choice)
**else**
  remove columns of $\widetilde{X}$ and $\widetilde{Z}$ not contained in $\mathcal{J}$
  **if** *outer optimum is feasible* **then**
    $w^* \leftarrow \arg\min_w w'\widetilde{Z}'\widetilde{Z}w$ s.t. $w \geq 0, \iota' w = 1$
    $v^* \leftarrow$ a particular $v$ with $w^* = W^*(v) = \arg\min_w w'\widetilde{X}'V(v)\widetilde{X}w$
  **else**
    **for** $k = 1, \ldots, K$ **do**
      Fix $v_k$ to 1
      $v^{(k)} \leftarrow \arg\min_v W^*(v)'\widetilde{Z}'\widetilde{Z}W^*(v)$ s.t. $v_k = 1, v \geq lb$
    $k^* \leftarrow \arg\min_k W^*(v^{(k)})'\widetilde{Z}'\widetilde{Z}W^*(v^{(k)})$
    $v^* \leftarrow v^{(k^*)}$
    $w^* \leftarrow W^*(v^*)$
  re-impute 0's to $w^*$ for components belonging to removed donors

Figure 1: Description of algorithm for generalized SCM problems

# 4 The R Package MSCMT

We illustrate the application of the R package MSCMT[10] by replicating some of the results of Abadie and Gardeazabal (2003). A more detailed and completely reproducible version of this section is included as a vignette called 'Working with package MSCMT' in package MSCMT[11].

[10] See Becker and Klößner (2016).

[11] See `https://cran.r-project.org/web/packages/MSCMT/vignettes/WorkingWithMSCMT.html`.

## 4.1 Preparing the Data

The R package MSCMT expects its input data to be a `list` of matrices, where each matrix corresponds to one variable of interest. The column names of the matrices correspond to the names of all units (treated unit and control units) and have to be identical across all elements of the list. The row names correspond to the instants of time where data is available, they may well be different between variables.

To facilitate using existing data collections, a helper function called `listFromLong`, which generates a list of matrices from a more common data representation, the so-called 'long' format, has been included in package MSCMT. With `listFromLong`, migrating from the Synth package to MSCMT is simple as well, because package Synth (more precisely, its `dataprep` function) expects its input data to be in 'long' format.

The `basque` dataset in package Synth in 'long' format is converted to the appropriate 'list' format for package MSCMT with the following commands:

```
library(Synth)
library(MSCMT)
data(basque)
Basque <- listFromLong(basque, unit.variable="regionno", time.variable="year",
                       unit.names.variable="regionname")
```

The 'list' representation allows for simple, reproducible data preparations. For the sake of completeness, we include the data transformations which are necessary to reproduce the analysis of Abadie and Gardeazabal (2003):

```
# define the sum of all cases
school.sum <- with(Basque, colSums(school.illit + school.prim + school.med +
                                   school.high  + school.post.high))
# combine school.high and school.post.high in a single class
Basque$school.higher <- Basque$school.high + Basque$school.post.high
# calculate ratios and multiply by number of observations to get percentages
for (item in c("school.illit", "school.prim", "school.med", "school.higher"))
  Basque[[item]] <- 5 * 100 * t(t(Basque[[item]]) / school.sum)
```

## 4.2 Defining the Model

The formulation of (M)SCM(T) models for application of package MSCMT consists of

- defining the (single)[12] treated unit (parameter `treatment.identifier`),
- defining the (multiple) control units (parameter `controls.identifier`),
- defining one or more dependent variables and the corresponding optimization periods as a `2 x L`-matrix (parameter `times.dep`), where
  - the column names consist of the names of the `L` dependent variables,
  - the first row contains the starting times of the optimization periods in the appropriate (annual, quarterly, or monthly) format,

[12] Changing the treated unit for placebo studies is done automatically.

- the second row contains the end times of the optimization periods in the appropriate format,

- defining several predictor variables and the corresponding optimization periods as a `2 x K`-matrix (parameter `times.pred`), where
  - the column names consist of the names of the `K` predictor variables,
  - the first row contains the starting times of the optimization periods in the appropriate (annual, quarterly, or monthly) format,
  - the second row contains the end times of the optimization periods in the appropriate format,

- an (optional) vector of the names of `K` aggregation functions (parameter `agg.fns`) for the predictor variables. If missing, all predictor variables are considered to be time series, which corresponds to the function name `"id"`. Whenever the result of an aggregation function has length exceeding 1, the resulting data is considered as a time series, too.

To reproduce the model specification of Abadie and Gardeazabal (2003), we define:

```
treatment.identifier <- "Basque Country (Pais Vasco)"
controls.identifier  <- setdiff(colnames(Basque[[1]]),
                                c(treatment.identifier, "Spain (Espana)"))
times.dep  <- cbind("gdpcap"                  = c(1960,1969))
times.pred <- cbind("school.illit"            = c(1964,1969),
                    "school.prim"             = c(1964,1969),
                    "school.med"              = c(1964,1969),
                    "school.higher"           = c(1964,1969),
                    "invest"                  = c(1964,1969),
                    "gdpcap"                  = c(1960,1969),
                    "sec.agriculture"         = c(1961,1969),
                    "sec.energy"              = c(1961,1969),
                    "sec.industry"            = c(1961,1969),
                    "sec.construction"        = c(1961,1969),
                    "sec.services.venta"      = c(1961,1969),
                    "sec.services.nonventa"   = c(1961,1969),
                    "popdens"                 = c(1969,1969))
agg.fns <- rep("mean", ncol(times.pred))
```

## 4.3 Estimation

After preparing the data and formulating the model, model estimation is done with function `mscmt`. As default, the genetic optimizer `genoud` of package `rgenoud` is used, and appropriate parameter defaults are chosen. Other optimizers or arbitrary overrides for the optimizer's parameters are supported.

In the following example, we use `DEoptim` of package `DEoptim`, another evolutionary optimizer, for the outer optimzation. Furthermore, we suppress the verbose output, which is active by default:

```r
res <- mscmt(Basque, treatment.identifier, controls.identifier, times.dep,
             times.pred, agg.fns, seed = 42, outer.optim = "DEoptim",
             verbose = FALSE)
```

The results of the estimation can be reported in human-readable form via the `S3` method for `print`, which is included in package MSCMT:

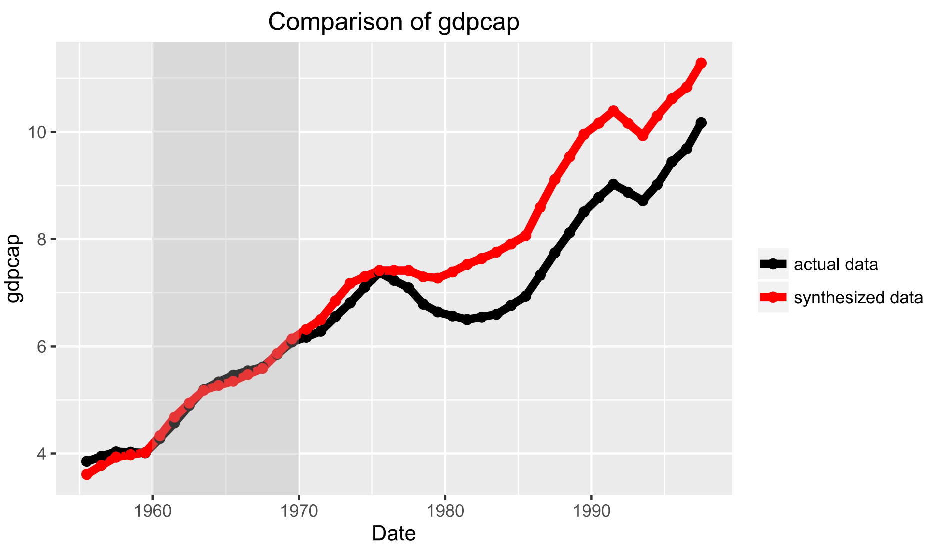
```r
res

## Specification:
## --------------
##
## Model type:     SCM
## Treated unit:   Basque Country (Pais Vasco)
## Control units:  Andalucia, Aragon, Principado De Asturias, Baleares (Islas),
##                 Canarias, Cantabria, Castilla Y Leon, Castilla-La Mancha, Cataluna,
##                 Comunidad Valenciana, Extremadura, Galicia, Madrid (Comunidad De),
##                 Murcia (Region de), Navarra (Comunidad Foral De), Rioja (La)
## Dependent(s):   gdpcap with optimization period from 1960 to 1969
## Predictors:     school.illit          from 1964 to 1969, aggregated via 'mean',
##                 school.prim           from 1964 to 1969, aggregated via 'mean',
##                 school.med            from 1964 to 1969, aggregated via 'mean',
##                 school.higher         from 1964 to 1969, aggregated via 'mean',
##                 invest                from 1964 to 1969, aggregated via 'mean',
##                 gdpcap                from 1960 to 1969, aggregated via 'mean',
##                 sec.agriculture       from 1961 to 1969, aggregated via 'mean',
##                 sec.energy            from 1961 to 1969, aggregated via 'mean',
##                 sec.industry          from 1961 to 1969, aggregated via 'mean',
##                 sec.construction      from 1961 to 1969, aggregated via 'mean',
##                 sec.services.venta    from 1961 to 1969, aggregated via 'mean',
##                 sec.services.nonventa from 1961 to 1969, aggregated via 'mean',
##                 popdens               from 1969 to 1969, aggregated via 'mean'
##
##
## Results:
## --------
##
## Result type:    Ordinary solution, ie. no perfect preditor fit possible and the
##                 predictors always impose some restrictions on the optimization of
##                 the 'dependent' loss.
## Optimal W:      Baleares (Islas)     : 21.93051%, Cataluna             : 63.27938%,
##                 Madrid (Comunidad De): 14.79011%
## Dependent loss: MSPE ('loss V'): 0.004286073,
##                 RMSPE           : 0.065468105
## (Optimal) V:    Some optimal weight vectors V are:
##                                                  min.loss.w    max.order
##                 school.illit.mean.1964.1969         9.998505e-09 1.661511e-05
##                 school.prim.mean.1964.1969          9.998505e-09 1.661511e-05
##                 school.med.mean.1964.1969           9.998505e-09 1.661511e-05
##                 school.higher.mean.1964.1969        9.998505e-09 3.054676e-04
##                 invest.mean.1964.1969               8.916218e-05 3.147949e-04
##                 gdpcap.mean.1960.1969               9.998505e-01 9.992136e-01
##                 sec.agriculture.mean.1961.1969      9.998505e-09 1.661511e-05
##                 sec.energy.mean.1961.1969           9.998505e-09 1.661511e-05
##                 sec.industry.mean.1961.1969         9.998505e-09 1.661511e-05
##                 sec.construction.mean.1961.1969     9.998505e-09 1.661511e-05
##                 sec.services.venta.mean.1961.1969   9.998505e-09 1.661511e-05
```

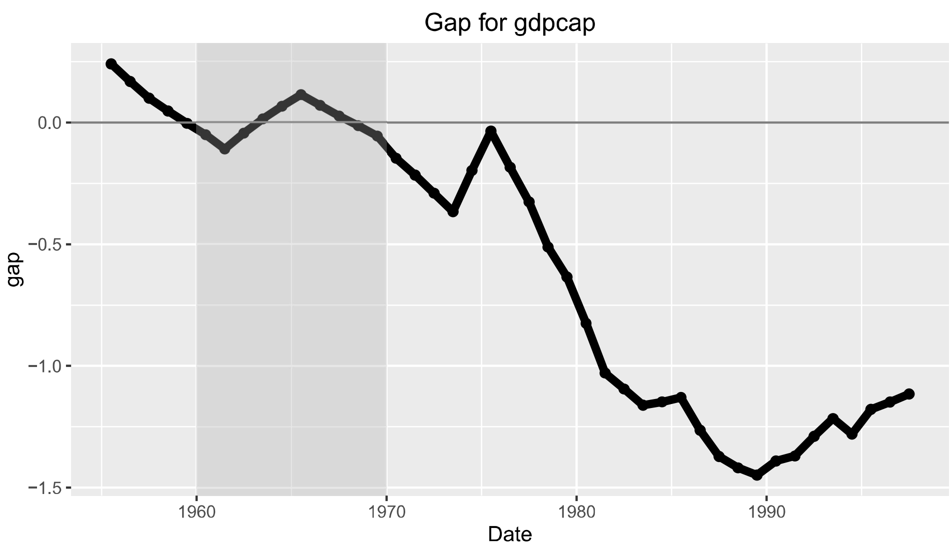
```
##                sec.services.nonventa.mean.1961.1969 6.020575e-05 1.661511e-05
##                popdens.mean.1969.1969               9.998505e-09 1.661511e-05
##                ----------
##                pred. loss                           1.199849e-04 3.552625e-04
##                (Predictor weights V are standardized by sum(V)=1)
##
```

For plotting, it is strongly recommended to use package `ggplot2` and the corresponding `S3` method for `ggplot` contained in package MSCMT. Comparison and gap plots are available for all (dependent and predictor) variables based on the results of a single estimation, the (first) dependent variable is chosen by default:

```
library(ggplot2)
ggplot(res, type="comparison")
```

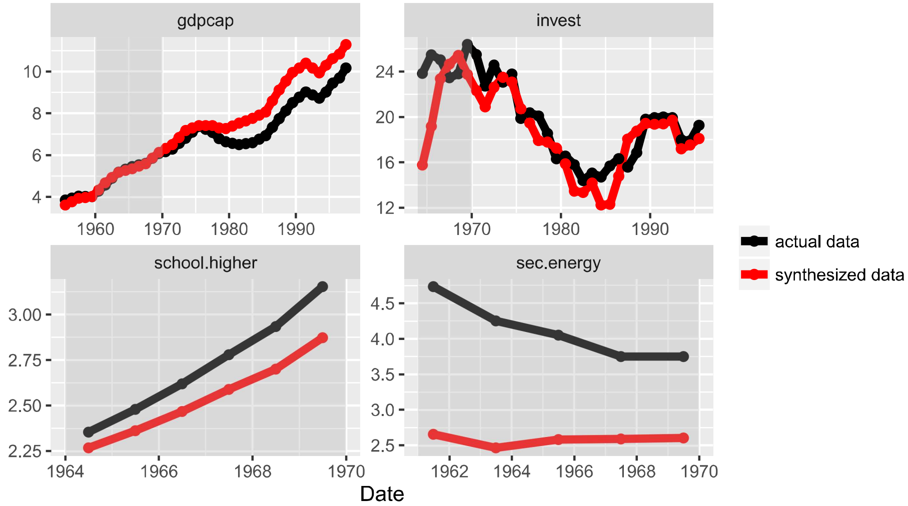



```
ggplot(res, type="gaps")
```




It is possible to plot several variables by providing a vector for argument `what`. Predefined sets of variables named `"dependents"`, `"predictors"`, and `"all"` can be selected with parameter `what.set`.

```r
ggplot(res, what=c("gdpcap","invest","school.higher","sec.energy"),
       type="comparison")
```




## 4.4 Placebo Study

Placebo studies are performed by simply setting the function argument `placebo` to `TRUE`. By default, the original treated unit is not added to the donor pool of the (original) control units, but this can be changed with parameter `placebo.with.treated`.

A remarkable speed-up (depending on the number of control units) can be achieved for placebo studies by making use of a cluster, which can be set up with the help of package `parallel`. The simplest form of a cluster is a local cluster, which makes the power of multi-core cpus available for the (lengthy) computations involved with placebo studies. Setting up a local cluster is very easy, see the example below.

The argument `cl` of function MSCMT can be used to specify the cluster to be used for placebo studies. The only drawback of using a cluster for placebo studies is losing the verbose output from the individual (one for each unit) SCM estimations, which includes a lack of progress information for the whole placebo study. Nevertheless, the speed-up should compensate for this drawback in all applications where a placebo study is meaningful.

In the following example, a (local) cluster with 6 nodes is used for the estimation.

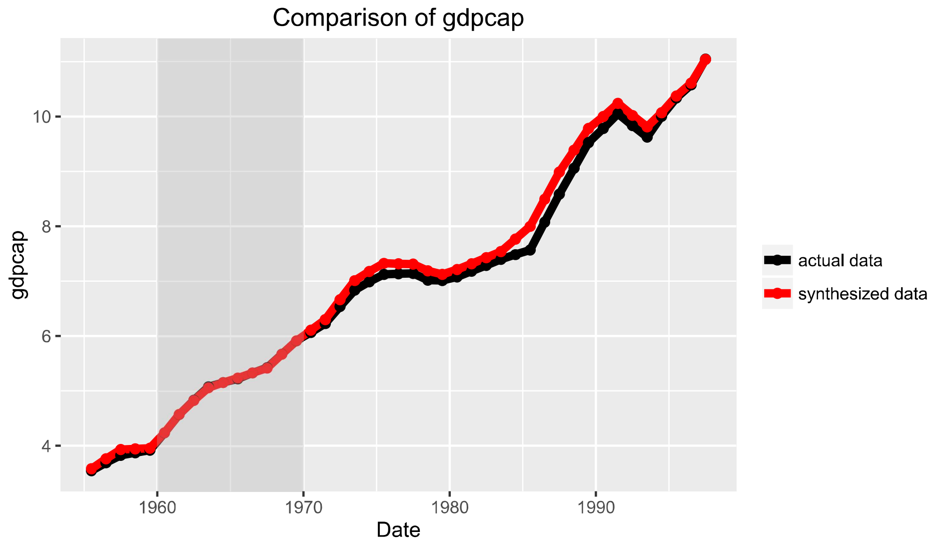
```r
library(parallel)
cl <- makeCluster(6)
resplacebo <- mscmt(Basque, treatment.identifier, controls.identifier,
                    times.dep, times.pred, agg.fns, cl=cl, placebo=TRUE,
                    seed=42, outer.optim="DEoptim")

## 15:50:49: Preparing cluster for placebo study.
## 15:50:50: Cluster prepared. Please hold the line.
## 15:50:50: Starting placebo study, excluding original treated unit.
## 15:51:48: Placebo study on cluster finished.

stopCluster(cl)
```

Object **resplacebo** now contains single SCM estimations for each unit as well as aggregated information concerning original data, synthesized data, and gaps for all units. The individual SCM estimations can be accessed separately (as list elements with names corresponding to the units' names). With the following plot, one can inspect whether there is some effect for Catalonia:

```
ggplot(resplacebo[["Cataluna"]], type="comparison")
```

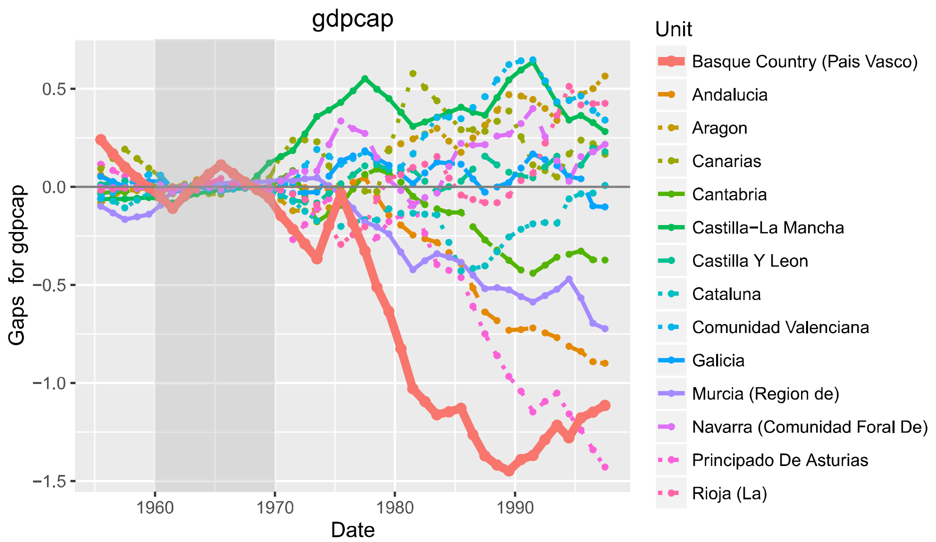



Several functions in package MSCMT are able to make use of the results of the placebo study as a whole. One example are so-called placebo plots, which is the default for plotting results of a placebo study.

Most often, not all control units can be synthesized with an acceptable fit in a placebo study, resulting in large pre-treatment gaps. Of course, large post-treatment gaps are expected for these units, but since these gaps are rather caused from lack of fit than from an existing treatment effect, excluding such units is strongly advisable while investigating the effect for the (original) treated unit.

In the following example, all control units with pre-treatment mspe of more than 5 times the treated unit's pre-treatment mspe are excluded from the placebo plot:

```
ggplot(resplacebo, exclude.ratio=5, ratio.type="mspe")
```



## 4.5 Statistical Inference

For statistical inference based on the results of a placebo study, the literature has developed so-called 'placebo tests', which have similarities to permutation tests. Two of these are

- tests for the (per-period) treatment effect based on the per-period gaps,
- tests for the aggregated treatment effect based on a difference-in-difference approach for aggregated (average) gaps.

Again, we exclude all control units with pre-treatment mspe of more than 5 times the treated unit's pre-treatment mspe.

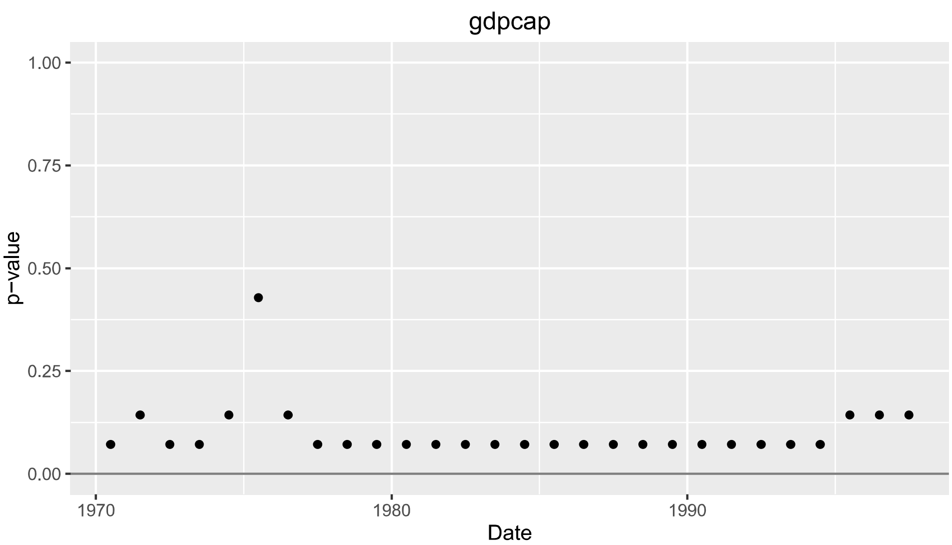
The p-values of per-period placebo tests can be calculated via function `pvalue`:

```
pvalue(resplacebo, exclude.ratio=5, ratio.type="mspe", alternative="less")

## Time Series:
## Start = 1970
## End = 1997
## Frequency = 1
##  [1] 0.07142857 0.14285714 0.07142857 0.07142857 0.14285714 0.42857143 0.14285714
##  [8] 0.07142857 0.07142857 0.07142857 0.07142857 0.07142857 0.07142857 0.07142857
## [15] 0.07142857 0.07142857 0.07142857 0.07142857 0.07142857 0.07142857 0.07142857
## [22] 0.07142857 0.07142857 0.07142857 0.07142857 0.14285714 0.14285714 0.14285714
```

A plot of these p-values can conveniently be created via `ggplot` when `plot.type` is set to `"p.values"`:

```
ggplot(resplacebo, exclude.ratio=5, ratio.type="mspe", type="p.value",
       alternative="less")
```




Calculating the aggregated treatment effect and testing its significance can be done with function `did` of package MSCMT, once again excluding all control units with pre-treatment mspe of more than 5 times the treated unit's pre-treatment mspe:

```r
did(resplacebo, range.post=c(1970,1990), exclude.ratio=5, alternative="less")

## $effect.size
## [1] -0.7715881
##
## $average.pre
## [1] 0.0006245798
##
## $average.post
## [1] -0.7709635
##
## $p.value
## [1] 0.07692308
##
## $rank
## [1] 1
##
## $excluded
## [1] "Baleares (Islas)"       "Castilla-La Mancha"    "Extremadura"
## [4] "Madrid (Comunidad De)"
```

Package MSCMT incorporates many more features in the current release, more detailed documentation and illustrations can be found in the package's vignettes and manual pages.

# 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 show how to detect two special cases overlooked in the literature and provide algorithms to solve the resulting SCM problems both numerically stable and fast. Furthermore, we show how the dimension of the problems to be solved may be reduced significantly by dropping irrelevant donor units, resulting in increased computation speed. We also elaborate on cleverly searching for synthetic controls when iterative optimizers have to be used for calculations, with special emphasis on numerical stability and computation speed.

We also showcase R package MSCMT to exemplify most of the newly developed routines as well as some of the many other features the package provides, like fast calculation of placebo studies by using parallel computing, handy plots, p values, and so on. For more details, for instance on handling multiple variables of interest or treating explanatory variables as time series, we refer the reader to the vignettes of package MSCMT, available at `https://cran.r-project.org/web/packages/MSCMT/vignettes/CheckingSynth.html`, `https://cran.r-project.org/web/packages/MSCMT/vignettes/UsingTimeSeries.html`, and `https://cran.r-project.org/web/packages/MSCMT/vignettes/WorkingWithMSCMT.html`.

## 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**, *MSCMT: Multivariate Synthetic Control Method Using Time Series* 2016. R package version 1.0.0.

**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.

**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.

**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.

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

**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.

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

# A Theory & Proofs

## A.1 Sunny Donors

**Definition 1.** *Given a set $\{\widetilde{x}_1, \ldots, \widetilde{x}_J\}$ of points in $K$-dimensional space with convex hull $H := \operatorname{conv}(\{\widetilde{x}_1, \ldots, \widetilde{x}_J\})$, we call $\widetilde{x}_{\widetilde{j}}$ ($\widetilde{j} = 1, \ldots, J$)* **shady** *(w.r.t. $H$ or w.r.t. $\widetilde{x}_1, \ldots, \widetilde{x}_J$) if there exists $0 < \alpha < 1$ and non-negative $\lambda_1, \ldots, \lambda_J$ with $\sum_{j=1}^{J} \lambda_j = 1$ and $\sum_{j=1}^{J} \lambda_j \widetilde{x}_j = \alpha \widetilde{x}_{\widetilde{j}}$. We call $x_{\widetilde{j}}$* **sunny** *(w.r.t. $H$ or w.r.t. $x_1, \ldots, x_J$) if it is not shady.*

**Proposition 1.** *1. Let $\widetilde{x}^* := \sum_{j=1}^{J} w_j^* \widetilde{x}_j$ be an optimizer of $\widetilde{x}' V \widetilde{x} \to \min$ in $H$ for which $(\widetilde{x}^*)' V \widetilde{x}^* > 0$. Then we have $w_{\widetilde{j}}^* = 0$ for all shady $\widetilde{x}_{\widetilde{j}}$.*

*2. $0 \in H$ if and only if no $x_{\widetilde{j}}$ is sunny.*

*Proof.* 1. Let $\widetilde{j}$ be such that $\widetilde{x}_{\widetilde{j}}$ is not sunny. Then there exist $\lambda_1, \ldots, \lambda_J \geq 0$ and $0 < \alpha < 1$ such that $\sum_{j=1}^{J} \lambda_j = 1$ and $\sum_{j=1}^{J} \lambda_j \widetilde{x}_j = \alpha \widetilde{x}_{\widetilde{j}}$. Using these, we define $\widetilde{\lambda}_j := \begin{cases} \frac{w_j^* + w_{\widetilde{j}}^* \frac{1}{\alpha} \lambda_j}{1 + w_{\widetilde{j}}^* \left(\frac{1}{\alpha} - 1\right)} & : \ j \neq \widetilde{j} \\ \frac{w_{\widetilde{j}}^* \frac{1}{\alpha} \lambda_{\widetilde{j}}}{1 + w_{\widetilde{j}}^* \left(\frac{1}{\alpha} - 1\right)} & : \ j = \widetilde{j} \end{cases}$, which are non-negative and sum to unity. We then have: $\frac{1}{1 + w_{\widetilde{j}}^* \left(\frac{1}{\alpha} - 1\right)} \widetilde{x}^* = \sum_{j=1}^{J} \widetilde{\lambda}_j \widetilde{x}_j$, because $\frac{1}{\alpha} \sum_{j=1}^{J} \lambda_j \widetilde{x}_j = \widetilde{x}_{\widetilde{j}}$. Therefore, $\frac{1}{1 + w_{\widetilde{j}}^* \left(\frac{1}{\alpha} - 1\right)} \widetilde{x}^* = \sum_{j=1}^{J} \widetilde{\lambda}_j \widetilde{x}_j$ lies in $H$, for which $\left(\frac{1}{1 + w_{\widetilde{j}}^* \left(\frac{1}{\alpha} - 1\right)} \widetilde{x}^*\right)' V \frac{1}{1 + w_{\widetilde{j}}^* \left(\frac{1}{\alpha} - 1\right)} \widetilde{x}^* = \frac{1}{\left(1 + w_{\widetilde{j}}^* \left(\frac{1}{\alpha} - 1\right)\right)^2} \left(\widetilde{x}^*\right)' V \widetilde{x}^*$. Thus, as $\widetilde{x}^*$ minimizes $\widetilde{x}' V \widetilde{x}$ in $H$, $1 + w_{\widetilde{j}}^* \left(\frac{1}{\alpha} - 1\right)$ must not exceed 1, which entails $w_{\widetilde{j}}^* = 0$.

2. '⇒' Let $H \ni 0 = \sum_{j=1}^{J} \alpha_j \widetilde{x}_j$ with $\alpha_1, \ldots, \alpha_J \geq 0$ and $\sum_{j=1}^{J} \alpha_j = 1$. For $\widetilde{j} \in \{1, \ldots, J\}$, define $\lambda_j := \begin{cases} \frac{1}{2} \alpha_j & : \ j \neq \widetilde{j} \\ \frac{1}{2}(\alpha_{\widetilde{j}} + 1) & : \ j = \widetilde{j} \end{cases}$. Then $\lambda_1, \ldots, \lambda_J \geq 0$, $\sum_{j=1}^{J} \lambda_j = 1$, and $\sum_{j=1}^{J} \lambda_j \widetilde{x}_j = \frac{1}{2} \widetilde{x}_{\widetilde{j}}$, i.e. $\widetilde{x}_{\widetilde{j}}$ is not sunny.

'⇐' By contradiction, assume that 0 was not in $H$. Then, let $V$ be the identity matrix and consider the minimization problem $\widetilde{x}'V\widetilde{x} = \widetilde{x}'\widetilde{x} \to \min$ in $H$. As $0 \notin H$, the optimizer $\widetilde{x}^* = \sum_{j=1}^{J} w_j^* \widetilde{x}_j$ fulfils $(\widetilde{x}^*)'\widetilde{x}^* > 0$. Then, due to the first part of the proposition, $w_j^* = 0$ for all $j$, entailing $\widetilde{x}^* = 0$, and we arrive at a contradiction.

□

## A.2 Finding Predictor Weights

**Proposition 2.** *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 conditions

$$Bw - v + \iota u = 0 \tag{7}$$

$$v'w = 0 \tag{8}$$

with $v = (v_1, \ldots, v_J) \geq 0$, $u \in \mathbb{R}$, are necessary and sufficient.[13] Multiplying equation (7) from left by $w'$ yields $u = -w'Bw$, which completes the proof. □

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