Appendix for: Policy Analysis using Synthetic Controls in Continuous-Time

Anonymous Authors[1]

This appendix provides additional material accompanying the main body of the paper: "Policy Analysis using Synthetic Controls in Continuous-time". It is outlined as follows:

- Section A presents proofs of the theoretical results presented in the main body of this paper.
  - Section A.1 provides an argument for the unbiasedness of treatment effects in continuous-time for a linear model, in analogy to (Abadie et al., 2010).
  - Section A.2 proves Proposition 1.
- Section B discusses the reliability of inference and computational complexity.
  - Section B.1 discusses the consistency of the recovered matrix $\mathbf{W}$ in different runs of the algorithm.
  - Section B.2 discusses the consistency of NC-SC's fit in different runs of the algorithm.
  - Section B.3 discusses computational complexity.
- Section C gives details on software and algorithm implementation.
- Section D gives further details on the design of experiments and gives pointers to the publicly available data and simulation environments.

A. Theoretical results

A.1. Unbiased treatment effects in linear dynamical systems

Synthetic controls and their use for treatment effect estimation is typically justified in the discrete-time setting by an underlying linear data generating mechanism involving observed, unobserved variables, and independent noise terms (Abadie et al., 2010). Extending this analysis to non-linear models is difficult and to our knowledge has not be done even in the discrete-time case.

We may provide some theoretical justification for the validity of NC-SC however by considering the continuous-time analog to the autoregressive linear model considered in (Abadie et al., 2010):

$$y_{i,t_{s+1}} = \alpha_t y_{i,t_s} + z_{i,t_s}, \quad \text{for} \quad i = 1, \dots, n \quad \text{and} \quad s = 1, \dots, m, \tag{1}$$

where $z_{i,t}$ is not modelled and may be correlated across units and time to account due to unobserved confounding but is assumed to have mean zero conditional on $\mathcal{F}_t = \{y_{i,t}\}_{i=1,\dots,n,t=t_1,\dots,t_m}$.

A continuous-time analog is given by

$$\frac{dy_i(t)}{dt} = \alpha(t) y_i(t) + z_i(t), \qquad y_i(t_0) = y_{i,0}, \tag{2}$$

for $t \in (t_0, t_m)$. All parameters and variables now vary continuously in time and the assumption, in parallel to equation (1) is that confounders $z_i(t)$ have mean zero conditional on $\mathcal{F}_t = \{y_i(s) : s < t\}_{i=1,\dots,n}$.

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

Following the steps taken in (Abadie et al., 2010) we may consider an infinitesimal change in time $\Delta t > 0$ and write

$$\frac{dy_i(t+\Delta t)}{dt} = \alpha(t+\Delta t)y_i(t+\Delta t) + z_i(t+\Delta t) \tag{3}$$

$$= \alpha(t+\Delta t)\left(y_i(t) + \Delta t \frac{dy_i(t)}{dt}\right) + z_i(t+\Delta t). \tag{4}$$

Assume now that there exists weights $w_2^\star, \ldots, w_n^\star$ such that for all times t before the intervention time T i.e., those times where we do have data and can compare the derivatives of the path of interest with control paths,

$$\frac{dy_1(t)}{dt} = \sum_{i=2}^{n} w_i^\star \frac{dy_i(t)}{dt}, \qquad y_{1,0} = \sum_{i=2}^{n} w_i^\star y_{i,0}, \qquad t < T. \tag{5}$$

It hols then that for $T = \Delta t + t$, for $\Delta t > 0$ and for $t < T$,

$$\frac{dy_1(T)}{dt} - \sum_{i=2}^{n} w_i^\star \frac{dy_i(T)}{dt} = \alpha(t+\Delta t)\left(y_1(t) - \sum_{i=2}^{n} w_i^\star y_i(t) + \Delta t\left(\frac{dy_1(t)}{dt} - \sum_{i=2}^{n} w_i^\star \frac{dy_i(t)}{dt}\right)\right) + z_1(T) - \sum_{i=2}^{n} z_i^\star y_i(T). \tag{6}$$

Then, using the relationship in (5), working recursively on the difference $y_1(t) - \sum_{i=2}^n w_i^\star y_i(t)$, and the fact that $z_1(T) - \sum_{i=2}^n z_i^\star y_i(T)$ has mean zero conditional on the filtration $\mathcal{F}_t$, the above quantity has mean zero.

We may then extend this result to $t > T$ by using equation (3).

(Abadie et al., 2010) also analyze factor models but since they do not incorporate time-varying confounders, observed or unobserved, we do not replicate their analysis here.

A.2. Proof of Proposition 1

We restate Proposition 1 for completeness, and redefine the two CDE models of interest.

We consider models of the form,

$$z_t = z_{t_0} + \int_{t_0}^{t} f(z_s)\, d\mathbf{Y}_s^0, \quad t \in (t_0, t_m], \tag{7}$$

and models of the form,

$$z_t = z_{t_0} + \int_{t_0}^{t} f(z_s)\, \mathbf{W} d\mathbf{Y}_s^0, \quad t \in (t_0, t_m], \tag{8}$$

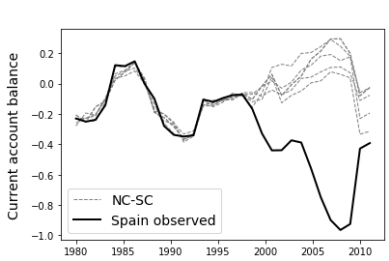
with the same notation as in the main body of this paper.

Proposition 1. *Consider the class of CDEs $\mathcal{C}$ defined by (7) that are independent of Y_i^0 and the class of CDEs $\mathcal{C}_0$ defined by (8) such that the i-th diagonal entry of $\mathbf{W}$ is zero. Then $\mathcal{C} = \mathcal{C}_0$.*

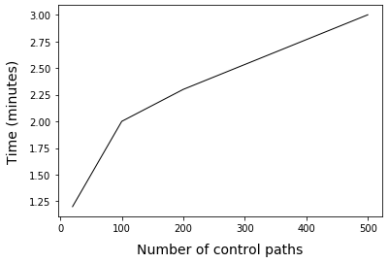
Proof. It is clear that the class of Neural CDEs $\mathcal{C}_0$ is contained in $\mathcal{C}$ as control paths interact with the vector field of the counterfactual path only if the corresponding entry in $\mathbf{W}$ is non-zero. $\mathcal{C}_0 \subset \mathcal{C}$.

For the converse consider a CDE in $\mathcal{C}$ defined by (7) independent of the k-th control path and consider a set of control paths $\tilde{\mathbf{Y}}_s^0$ and $\mathbf{Y}_s^0$ such that $\tilde{\mathbf{Y}}_s^0 = \mathbf{Y}_s^0$ except for the k-th control path of $\tilde{\mathbf{Y}}_s^0$ which is set to the zero function $\tilde{Y}_k^0 : [t_0, t_m] \to 0$, for all $s \in [t_0, t_m]$. Because of independence, the CDEs with control paths $\tilde{\mathbf{Y}}_s^0$ and $\mathbf{Y}_s^0$ are equal.

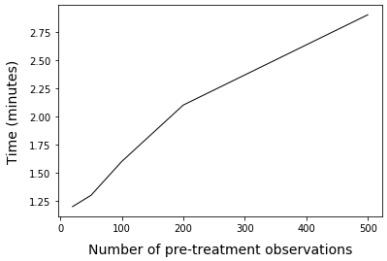
Define the diagonal matrix $\tilde{\mathbf{W}}$ and $\mathbf{W}$ such that their diagonal entries agree except on their k-th diagonal entry of $\tilde{\mathbf{W}}$ where $[\tilde{\mathbf{W}}]_{kk} = 0$. Then we can see that $\tilde{\mathbf{W}} d\mathbf{Y}_s^0 = \mathbf{W} d\tilde{\mathbf{Y}}_s^0$, so that the two CDEs given by (8) define the same functions. A CDE (7) independent of one of its arguments may always be reconstructed with a corresponding CDE (8). This implies that the class of Neural CDEs independent of the k-th control path is contained in $\mathcal{C}_0$. $\mathcal{C} \subset \mathcal{C}_0$ □



(a) NC-SC fit over 5 runs with different initializations on the current account deficit data for Spain.



(b) Run time of NC-SC as a function of the number of control paths.



(c) Run time of NC-SC as a function of the number of pre-treatment observations.

Figure 1. Analysis of fit consistency and computational complexity.

B. Additional experiments

B.1. Consistency of W

Different runs of the algorithm may converge to different local minima thus potentially changing the interpretation of the matrix $\mathbf{W}$ of control path contributions. We did observe slight differences in the entries of the estimated matrix $\mathbf{W}$ although its interpretation: which entries were estimated to zero and non-zero, remained consistent.

We tested this feature with the current account deficit data. We ran NC-SC 10 times with different initializations and report the variance of estimated weights for each country in Table 1. For almost all entries the weight variance is consistently low and no run gives a qualitatively different interpretation (for instance one run estimating $\mathbf{w}_i$ to zero while another run estimating it to a value different than zero).

While this is not an exhaustive test of this behaviour it provides some evidence that $\mathbf{W}$ may be used consistently to recover the control paths most influential in counterfactual estimation.

CHL	DNK	HUN	ISR	JPN	MEX	NZL	POL	SWE	USA	Others
.127 (.02)	.063 (.01)	.076 (.01)	.068 (.02)	.123 (.04)	.105 (.03)	.051 (.02)	.096 (.02)	.089 (.01)	.088 (.02)	.000 (.00)

Table 1. Weight estimation over 10 runs of NC-SC. Others are: Canada, Turkey, Great Britain, Australia and Korea.

B.2. Consistency of NC-SC fit

Just as different runs of NC-SC may produce different estimated matrices $\mathbf{W}$, may result in different counterfactual fits. Using the same experiment as above we show that NC-SC's fit is remarkably consistent across different initializations.

Figure 1 shows the fit of NC-SC in 5 runs with different initialization parameters. We used the current account deficit data for this experiment.

B.3. Computational complexity

Run times of NC-SC as a function of the number of pre-treatment observations and as a function of the number of control paths is given in Figure 1. We used the Lorenz model for this experiment.

C. Details on algorithm implementation

This section gives details on the implementation of our algorithm as well as implementation software of baseline methods.

C.1. Neural Continuous Synthetic Controls

For completeness, this section reviews our modelling choices for Neural CDEs and alternatives, following the analysis of (Kidger et al., 2020).

NC-SC is defined using the Neural CDE,

$$z_t = z_{t_0} + \int_{t_0}^{t} f(z_s)\ \mathbf{W}\ d\mathbf{Y}_s^0, \quad t \in (t_0, t_m], \tag{9}$$

with the notation used in the main body of this paper.

- **Path interpolation.** We followed (Kidger et al., 2020) in approximating the underlying paths $\mathbf{Y}_s^0$ using cubic spline interpolations with knots at the observation times. The minimum requirement for evaluating the CDE in equation (9) is therefore that $\mathbf{Y}_s^0$ be at least continuous and piece-wise differentiable.

  However, training with the adjoint backpropagation method requires derivatives of a functional of the CDE with respect to time, an thus second derivatives of the path approximations $\mathbf{Y}_s^0$. For this to be done consistently, the choice of cubic splines essentially gives the minimum smoothness requirement to paths approximations.

- **Other vector field choices.** If one wanted to incorporate the influence of auxiliary paths to modulate the trajectory of z, different choices for the vector field "$f(z_s)\ d\mathbf{Y}_s^0$" could have been made. For instance, (De Brouwer et al., 2019) developed a time series method defining the vector field as $g(z_s, Y_s^0)$ for some function g to be learned. This is perhaps a more natural choice that considers a function of paths $\mathbf{Y}_s^0$ explicitly and allows for non-linearities in the interaction between z and $\mathbf{Y}_s^0$. (Kidger et al., 2020) showed, however, that the vector field "$f(z_s)\ d\mathbf{Y}_s^0$" is strictly more general, subsuming models of the form $g(z_s, Y_s^0)$.

- **Architecture.** The integrand f was taken to be a feed-forward neural network with a two hidden layers of size 10 and elu activation functions after each layer except from the output layer. The dimensionality l of z the hidden state was taken to be 5 for all experiments. The activation function was chosen to be the `elu` function, although the `relu` performed similarly but was less stable in optimization with a larger variance across different runs of the algorithm.

  We did not explicitly tune hyperparameters (hidden layers, activation function, etc.) for performance. Further tuning could be done by cross-validation on the observed pre-treatment trajectory of the unit of interest if enough observations are given.

- **Optimization.** In each case we used the Adam optimiser as implemented by PyTorch. Starting learning rates varied between experiments (with values between 0.001 and 0.01) before being reduced by half if metrics failed to improve for a certain number of epochs.

  The strength of $\mathbf{W}$ regularization λ is chosen in a range $\{0.001, 0.01, 0.1, 1\}$ for best performance on a validation set.

  The ODE solver used to extrapolate the hidden state was taken to be the fourth-order Runge-Kutta with 3/8 rule solver, as implemented by passing `method='rk4'` to the `odeint_adjoint` function of the `torchdiffeq` (Kidger et al., 2020; Chen et al., 2018) package, used also for adjoint back-propagation training. The step size was taken to equal the minimum time difference between any two adjacent observations.

C.2. Original Synthetic Controls

We implement the original synthetic control method (Abadie et al., 2010) with the `cvxpy` python package for constrained convex optimization. This allows us to enforce weights to be non-negative and sum to one in few straightforward lines of code.

C.3. Robust Synthetic Controls

We implement a penalized version of the original synthetic control method with an elastic net penalty and hyperparameters chosen by cross-validation with the `sklearn` python library and predefined optimization function `ElasticNetCV`.

C.4. Synthetic Controls with Kernel Mean Matching

We implement a modification of the original synthetic control method, instead matching weights in a reproducing kernel Hilbert space with Gaussian kernel and bandwidth parameter taken to be the median distance between any pair of observations as suggested by the authors (Gretton et al., 2009). Our implementation code follows exactly the procedure given in the constrained optimization program defined in section 1.1.3 of (Gretton et al., 2009).

The algorithm itself was implemented in python with a constrained quadratic program `cvxpy`, similarly to the original synthetic control method.

C.5. Matrix Completion

We implement the matrix completion method with nuclear norm penalization of (Athey et al., 2018) with the `matrix completion` package in python. Its implementation is straightforward with this package taking only a single line of code to define and fit the model.

D. Experiment details

D.1. Smoking and Eurozone experiments

The data from both the smoking study in California and current account deficits in Spain is annualized. For a period of 30 years we thus have 30 observations on each trajectory (e.g. cigarette sales in California, Nevada, etc., and current account deficits in Spain, Switzerland, etc.) and in particular in both studies we have 19 years / observations of pre-intervention data that we may use for quantitative performance evaluations.

This is rarely sufficient for reliable comparisons between algorithms. Our approach to correct for this problem is to augment the data by interpolating trajectories with cubic splines with knots at the observation times. The smoothness hyperparameter is chosen such that the interpolation accurately reflects the original data trajectory. With this interpolation, we evaluate the curve for each path at 300 regular points spanning the 19 years of pre-intervention data.

In our experiment we use the first 200 observations for training and we use the last 100 observations for testing, on which we report performance.

References

Abadie, A., Diamond, A., and Hainmueller, J. Synthetic control methods for comparative case studies: Estimating the effect of california's tobacco control program. *Journal of the American statistical Association*, 105(490):493–505, 2010.

Athey, S., Bayati, M., Doudchenko, N., Imbens, G., and Khosravi, K. Matrix completion methods for causal panel data models. Technical report, National Bureau of Economic Research, 2018.

Chen, R. T., Rubanova, Y., Bettencourt, J., and Duvenaud, D. K. Neural ordinary differential equations. In *Advances in neural information processing systems*, pp. 6571–6583, 2018.

De Brouwer, E., Simm, J., Arany, A., and Moreau, Y. Gru-ode-bayes: Continuous modeling of sporadically-observed time series. In *Advances in Neural Information Processing Systems*, pp. 7379–7390, 2019.

Gretton, A., Smola, A., Huang, J., Schmittfull, M., Borgwardt, K., and Schölkopf, B. Covariate shift by kernel mean matching. *Dataset shift in machine learning*, 3(4):5, 2009.

Kidger, P., Morrill, J., Foster, J., and Lyons, T. Neural controlled differential equations for irregular time series. *arXiv preprint arXiv:2005.08926*, 2020.