Demonstrating an illusory reduction in crime via sampling

James Macdonald, Data Scientist, MDSc

November 9, 2020

1 Introduction

The following is a critical review of "An evaluation of the Suspect Target Management Plan" (the report), published by the NSW Bureau of Crime Statistics and Research (BOCSAR) in October 2020. The report attempts to measure the impact of the Suspect Target Management Plan (STMP) on individual criminal behaviour. This critique will concentrate on confounding bias, and demonstrate how the reports conclusions are consistent with a nil-treatment-effect scenario.

The study hinges on an observed reduction in criminal activity after placement on an STMP compared to the year leading up to an STMP. The author has done an admirable job, working with a difficult dataset, and attempted to control for detection and selection bias by concentrating purely high-visibility crimes with a greater likelihood of reporting.

Unfortunately, there is still critical, unaddressed confounding bias, which undermines the conclusions of the study. By using people pre-STMP individuals as a control group for those post-STMP, the analysis overlooks the inherent relationship between the dependent variable, offending, and the treatment variable, STMP. The construction of the dataset introduces a confounding bias by observing individual behaviour conditioned on commencement of an STMP.

Via the below experiment, we demonstrate how an observation window that depends on the variable being measured can introduce a confounding bias into the data, and how any subsequent models will also be biased.

Concisely, this will demonstrate that for a given event E, time t and experiment commencement time T, the dataset will satisfy the inequality:

$$\frac{P(E|t<T)}{P(E|t>T)} > 1$$

if the observation point depends on such an event occuring. We will demonstrate that this inequality occurs even when individual behaviour is unchanged between target and control. Futher, we demonstrate that this will produce a bias in a subsequent models.

[1]:
```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
from scipy import stats
from sklearn.linear_model import LogisticRegression
import statsmodels.api as sm
```

```
%matplotlib inline
```

1.1 Assumptions

We take 100,000 persons have individually varying criminal tendencies. We express this by articulating a threshold beyond which an individual commits a crime, drawn from a highly-skewed beta distribution and denoted by crim_thresh. We draw a similar STMP commencement threshold stmp_thresh from a uniform distribution. If a crime occurs, it lowers the probability of evading STMP by $\frac{1}{1+(n_c * f_c)}$ where n_c is the count of crimes and f_c is a severity factor.

[2]:
```
%%time
np.random.seed(1)

n_ind = 100000 # Simulate 100K individuals
p_stmp = 0.0001 # Given no crime, individual's probability of being put on a␣
↪STMP on a given day
crime_thresh = stats.beta(a=100,b=1).rvs(size=n_ind) # For each individual, set␣
↪a daily crime factor via beta dist.
stmp_thresh = stats.uniform().rvs([n_ind,1000]) # For each individual, set a␣
↪daily stmp threshold.
decay = 200 # how long it takes chance of STMP to revert
crime_mult = 2 # Factor-decrease of chances of evading STMP from crime:␣
↪Prob~(1-1/ (fctr*crime))
```

```
CPU times: user 990 ms, sys: 108 ms, total: 1.1 s
Wall time: 1.1 s
```

1.2 Experiment

We then assign everyone an activity score from a uniform distribution, and where this is greater than their individual crime threshold, this becomes a crime.

Next, we calculate an STMP evasion score, and apply a base score that someone without criminal activity receives an STMP.

Where someone's STMP evasion score falls below their STMP threshold, they are assigned an STMP. We then assign that individual an observation point based on their commencement of STMP, and remove individuals who never receive an STMP.

Figure 1 shows that this data gives us the desired result - varied criminal behaviour that is positively correlated with the probability of receiving an STMP, and a decay rate of STMP likelihood post criminal behaviour.

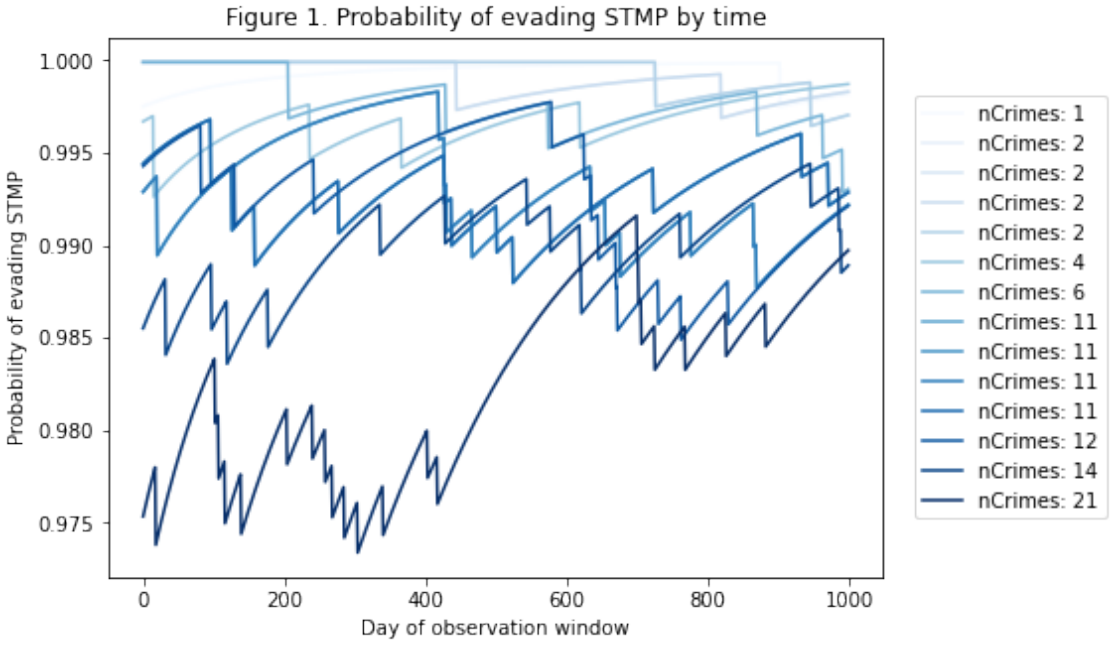
[3]:
```
%%time
activity = stats.uniform().rvs([n_ind,1000+decay]) # Roll dice on crime␣
↪activity for all individuals for 1000 days
crime = pd.DataFrame(activity.T > crime_thresh).astype(int) # crime = activity␣
↪> individual crime threshold
```

```
stmp_prop = (1/(1+crime_mult*crime) - p_stmp).ewm(halflife=decay).mean() #␣
 ↪probability of STMP given crime

crime.columns = [str(i) for i in crime.columns.values] # rename columns
stmp_prop = stmp_prop[decay:].reset_index(drop=True) # remove padding
crime = crime[decay:].reset_index(drop=True) # remove padding

stmps = (stmp_prop < stmp_thresh.T).astype(int) # STMP occurs where probability␣
 ↪> daily threshold
obs_point = stmps.idxmax() # Observation point is commencement of STMP
obs = obs_point[(obs_point>365)&(obs_point<1000-365)] # trim where stmp occurs␣
 ↪near edge of observation window
stmp_clean = stmps[obs.index]

observed_crime = pd.DataFrame()
for icol in stmp_clean.columns.values:
    observed_crime[icol] = crime[str(icol)][obs_point.loc[icol]-365:obs_point.
 ↪loc[icol]+365].values
```

```
CPU times: user 9.55 s, sys: 1.9 s, total: 11.5 s
Wall time: 11.5 s
```

[4]:
```
fig, ax = plt.subplots(figsize=(7,5))
to_plot = stmp_prop[stmp_prop.columns[:10]]
to_plot.columns = crime[crime.columns[:10]].sum().values
to_plot = to_plot[sorted(to_plot.columns)]
to_plot.columns = [f'nCrimes: {i}' for i in to_plot.columns.values]


to_plot.plot(ax=ax, cmap='Blues')
ax.set_title("Figure 1. Probability of evading STMP by time")
ax.set_xlabel("Day of observation window")
ax.set_ylabel("Probability of evading STMP")
ax.legend(loc='right', bbox_to_anchor=(1.3,0.5))
```

[4]:
```
<matplotlib.legend.Legend at 0x7f693cc696d0>
```

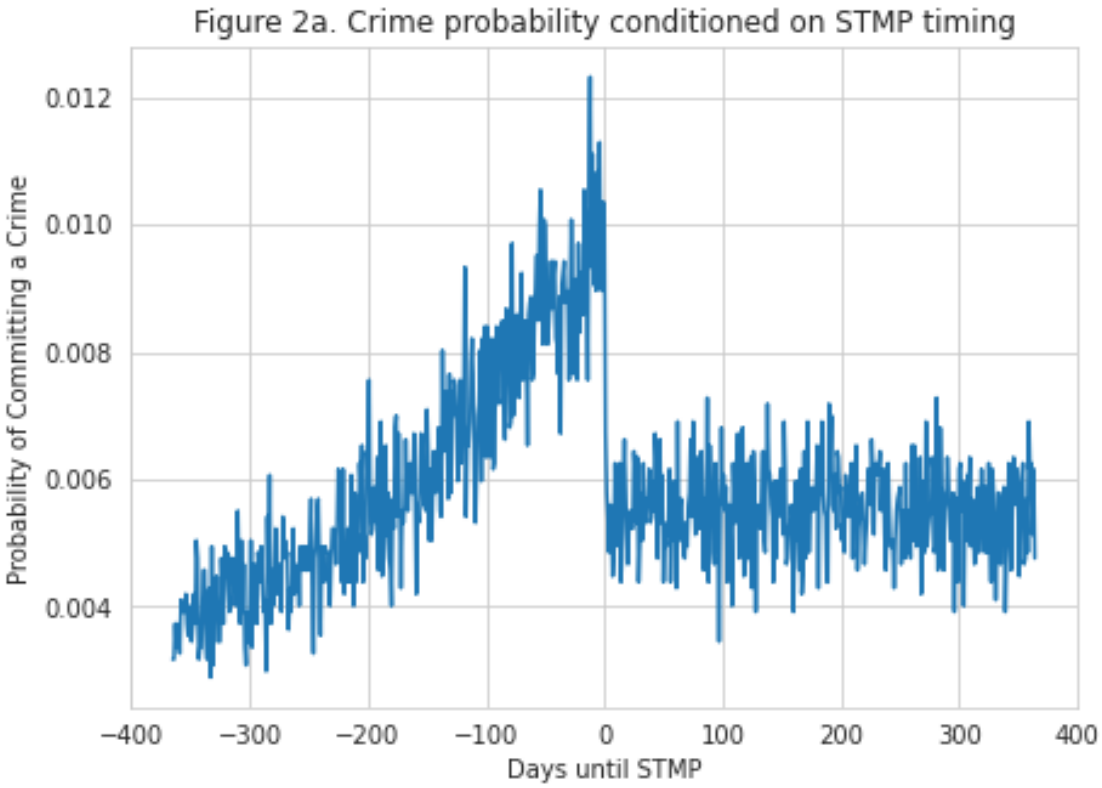


1.3 Results

We can see below that the data shows a rise in criminal activity preceding the observation point, followed by a dramatic drop. This plot is very similar to figure 1a of the report, which plots the same data.

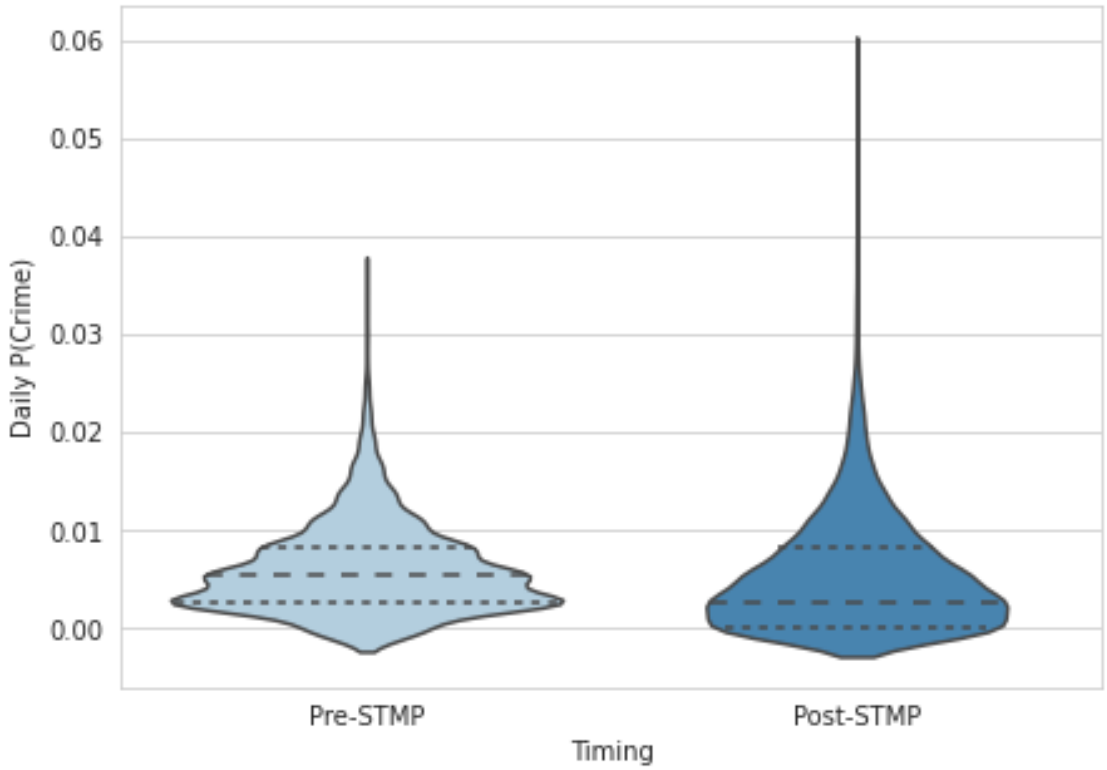
[5]:
```
sns.set_style('whitegrid')
fig, ax = plt.subplots(figsize=(7,5))
observed_crime.set_index(np.arange(-365,365)).mean(axis=1).plot(ax=ax)
ax.set_xlabel("Days until STMP")
ax.set_ylabel("Probability of Committing a Crime")
ax.set_title("Figure 2a. Crime probability conditioned on STMP timing")
```

[5]:
```
Text(0.5, 1.0, 'Figure 2a. Crime probability conditioned on STMP timing')
```



[6]:
```
fig, ax = plt.subplots(figsize=(7,5))
cols = ['Pre-STMP','Post-STMP']
pre_and_post = pd.DataFrame([observed_crime[:365].mean(),observed_crime[365:].
 ↪mean()], index = cols).T
for_vplot = pd.melt(
    frame= pre_and_post, value_vars = ['Pre-STMP','Post-STMP'],␣
 ↪value_name='Daily P(Crime)', var_name='Timing'
)
sns.violinplot(data=for_vplot, x='Timing', y='Daily P(Crime)', palette='Blues',␣
 ↪inner='quartile', bw=0.25, ax=ax)
```

[6]:
```
<AxesSubplot:xlabel='Timing', ylabel='Daily P(Crime)'>
```



[7]:
```
pre_and_post.describe()[1:].apply(lambda x: x.apply(lambda y:␣
↪f'{round(y*100,4)}%'))
```

[7]:

	Pre-STMP	Post-STMP
mean	0.6063%	0.5531%
std	0.4579%	0.561%
min	0.0%	0.0%
25%	0.274%	0.0%
50%	0.5479%	0.274%
75%	0.8219%	0.8219%
max	3.5616%	5.7534%

[8]:
```
def calculate_trimmed_mean(qvalue):
    pre_outliers = pre_and_post['Pre-STMP'] < pre_and_post['Pre-STMP'].
↪quantile(qvalue)
    post_outliers = pre_and_post['Post-STMP'] < pre_and_post['Post-STMP'].
↪quantile(qvalue)
    return pd.concat([
    pre_and_post['Pre-STMP'][pre_outliers].describe(),
    pre_and_post['Post-STMP'][post_outliers].describe()
], axis=1)
```

```
calculate_trimmed_mean(0.96)[1:].apply(lambda x: x.apply(lambda y:␣
↪f'{round(y*100,3)}%'))
```

[8]:

	Pre-STMP	Post-STMP
mean	0.541%	0.451%
std	0.353%	0.402%
min	0.0%	0.0%
25%	0.274%	0.0%
50%	0.548%	0.274%
75%	0.822%	0.822%
max	1.37%	1.37%

1.4 Interpretation

The above plots and summary statistics demonstrate that, despite zero change in underlying individual behaviour, there is a dramatic observed shift in Pre-STMP and Post-STMP criminal activity. This shift can be described as the function of the inequality between conditional probabilities, from the above table we can substitute our results and maintain the inequality:

$$\frac{P(E|t<T)}{P(E|t>T)} = \frac{\mathbf{E}[Crime_{preSTMP}]}{\mathbf{E}[Crime_{postSTMP}]} = \frac{0.61\%}{0.55\%} > 1$$

Furthermore, the median post-STMP subject has double the offending likelihood of the median pre-STMP subject. Trimmed summary statistics that exclude upper tails would further exacerbate the difference between mean values. Excluding the top 4% of outliers increases mean treatment effect from 9% to 17%.

1.5 Effect on modelling

There is no magic mechanism in modelling that can overcome this bias. Passing the data through a model will necessarily return a negative coefficient due to the relationship between the treatment and the outcome. This is demonstrated below, with a logistic regression returning a coefficient of β_{stmp} = -5.19, with a p-value significant at any level.

[9]:
```
is_stmp = pd.DataFrame(np.zeros_like(observed_crime))
is_stmp.loc[np.arange(365,730)] = 1
```

[10]:
```
X = pd.DataFrame(is_stmp.stack())
y = observed_crime.stack()
```

[11]:
```
%%time
lreg = sm.Logit(y.values, X.values).fit()
print(lreg.summary())
```

```
Optimization terminated successfully.
         Current function value: 0.363705
         Iterations 9
                           Logit Regression Results
```

```
==============================================================================
Dep. Variable:                      y   No. Observations:              7822680
Model:                          Logit   Df Residuals:                  7822679
Method:                           MLE   Df Model:                            0
Date:                Mon, 09 Nov 2020   Pseudo R-squ.:                 -9.205
Time:                        10:57:30   Log-Likelihood:            -2.8451e+06
converged:                       True   LL-Null:                   -2.7879e+05
Covariance Type:            nonrobust   LLR p-value:                       nan
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
x1            -5.1918      0.007   -761.524      0.000      -5.205      -5.178
==============================================================================
CPU times: user 41.2 s, sys: 18.4 s, total: 59.6 s
Wall time: 24.5 s
```

Furthermore, including control variables to limit confounding will not address this bias. In order to have a meaningful effect on the estimation of β_{stmp} a control variable would necessarily be correlated with the STMP variable in the data, and this has been explicitly ruled out in the report by attempting to match on proper controls. We can assume that no control variables are correlated with treatment, and thus no control variables can remove the bias in the model estimates.

1.6 Conclusion

This experiment demonstrates that an observation time triggered by a treatment variable that is correlated with the dependent variable introduces a bias into the dataset. The observed distributions of pre-treatment and post-treatment behaviour are significantly different despite, *by construction*, no change in underlying individual behaviour. It also demonstrates that the bias in the dataset will produce spurious model coefficients, and this cannot be rectified by the introduction of control variables.

In the context of the report, using pre-STMP individuals as controls for post-STMP individuals would carry the issues identified above. Furthermore, no dataset constructed from the subset of individuals on STMP is able to provide meaningful insight into the impact of STMPs, irrespective of the modelling or matching methodology used. This could be rectified by matching against individuals who are not subject to STMP in order to provide a counterfactual.