Supplementary Software

The following statistical code was used for analyses of the New Zealand nationwide administrative data.

```
/*************************************************************************
*************************************************************************
**********
AUTHOR: Stephanie D'Souza
IDI REFRESH: idi_clean_20171020

PURPOSE:
- code for deriving each cohort and total population

DISCLAIMER:
The results in this [report, paper] are not official statistics.
They have been created for research purposes from the Integrated Data
Infrastructure (IDI), managed by Statistics New Zealand.
The opinions, findings, recommendations, and conclusions expressed in this
[report, paper etc] are those of the author(s), not Statistics NZ,
[Department X, or Organisation Y].
Access to the anonymised data used in this study was provided by Statistics
NZ under the security and confidentiality provisions of the Statistics Act
1975.
Only people authorised by the Statistics Act 1975 are allowed to see data
about a particular person, household, business, or organisation,
and the results in this [report, paper] have been confidentialised to protect
these groups from identification and to keep their data safe.
Careful consideration has been given to the privacy, security, and
confidentiality issues associated with using administrative and survey data
in the IDI.
Further detail can be found in the Privacy impact assessment for the
Integrated Data Infrastructure available from www.stats.govt.nz.

*************************************************************************
*************************************************************************
*********/

* specify libraries;
libname central ODBC dsn=idi_clean_20171020_srvprd schema=data;
libname dia ODBC dsn=idi_clean_20171020_srvprd schema=dia_clean;
libname data ODBC dsn=idi_clean_20171020_srvprd schema=data;
libname central ODBC dsn=idi_clean_20171020_srvprd schema=data;
libname steph "/nas/DataLab/MAA/MAA2018-15 A small segment of the population
with large economic burden/Steph/Data";

** create sample;
data total_pop;
set dia.births;
if 1950 <= dia_bir_birth_year_nbr <= 1984;
keep snz_uid dia_bir_birth_month_nbr dia_bir_birth_year_nbr;
run;

* remove duplicates;
proc sort data=total_pop nodupkey;
```

```
by snz_uid dia_bir_birth_month_nbr dia_bir_birth_year_nbr;
run;

* exclude those not in spine (data.personal_detail is spine table), and get
sex code;
proc sql;
      create table total_pop_1 as
      select a.*, b.snz_sex_code from total_pop a
      left join data.personal_detail b
            on a.snz_uid = b.snz_uid
            where b.snz_spine_ind = 1;
quit;

* merge cohorts with deaths;
proc sql;
      create table totpop as
            select a.*, b.dia_dth_death_month_nbr, b.dia_dth_death_year_nbr
from total_pop_1 a
            left join dia.deaths b
                  on a.snz_uid = b.snz_uid;
quit;

*remove duplicates;
proc sort data=totpop nodupkey;
by snz_uid;
run;

************************************************************;
** overseas indicators - code below lets us calculate the number of days
overseas, adapted from C. Liu;
* does this for six month blocks between 01 July 2006 - 31 June 2016;
* create cohort with days overseas info;
%macro os(
  thedata,
  start_yr /* first year to produce estimates for */,
  end_yr /* final year to produce estimates for */);


data pop;
set &thedata;
FORMAT DOB DOD DATE10.;
dob=MDY(dia_bir_birth_month_nbr,1,dia_bir_birth_year_nbr);
dod=MDY(dia_dth_death_month_nbr,1,dia_dth_death_year_nbr);
sex=snz_sex_code;
keep snz_uid dob dod sex;
run;

%do year = &start_yr. %to &end_yr.;

******************
  ** first half ***
  *****************;

proc sql;
create table overseas_spells as
select distinct a.snz_uid , dob, dod, sex, pos_applied_date, pos_ceased_date,
pos_day_span_nbr
```

```
from pop a
left join
(select snz_uid, pos_applied_date, pos_ceased_date, pos_day_span_nbr
  from central.person_overseas_spell
  where pos_applied_date <= "30JUN&year.:23:59:59.999"dt and pos_ceased_date
>= "01JAN&year.:00:00:00.000"dt) b
on a.snz_uid=b.snz_uid
order by snz_uid, pos_applied_date;
quit;


** Calculate number of days spent overseas **;
data overseas_time;
set overseas_spells;
if pos_ceased_date > "30JUN&year.:23:59:59.999"dt and pos_applied_date <
"01JAN&year.:00:00:00.000"dt
then time_to_add = 181;

else if pos_ceased_date > "30JUN&year.:23:59:59.999"dt and pos_applied_date
>= "01JAN&year.:00:00:00.000"dt
then time_to_add = ("30JUN&year.:23:59:59.999"dt - pos_applied_date) / 86400;

else if pos_ceased_date <= "30JUN&year.:23:59:59.999"dt and pos_applied_date
>= "01JAN&year.:00:00:00.000"dt
then time_to_add = (pos_ceased_date - pos_applied_date) / 86400;

else if pos_ceased_date <= "30JUN&year.:23:59:59.999"dt and pos_applied_date
< "01JAN&year.:00:00:00.000"dt
then time_to_add = (pos_ceased_date - "01JAN&year.:00:00:00.000"dt) / 86400;

if missing(pos_day_span_nbr) then time_to_add = 0;
run;

proc sql;
create table time_overseas_&year._1 as
select snz_uid, dob, dod, sex,
ROUND(SUM(time_to_add),1) as days_overseas
from overseas_time
group by snz_uid, dob, dod, sex;

create table time_overseas_&year._1 as
select *,
"01JAN&year."d as refdatestart format=date10.,
"30JUN&year."d as refdateend format=date10.
from time_overseas_&year._1;

quit;


******************
  ** second half ***
  *****************;

proc sql;
create table overseas_spells as
select distinct a.snz_uid , dob, dod, sex, pos_applied_date, pos_ceased_date,
pos_day_span_nbr
```

```
from pop a
left join
(select snz_uid, pos_applied_date, pos_ceased_date, pos_day_span_nbr
  from central.person_overseas_spell
  where pos_applied_date <= "31DEC&year.:23:59:59.999"dt and pos_ceased_date
>= "01JUL&year.:00:00:00.000"dt) b
on a.snz_uid=b.snz_uid
order by snz_uid, pos_applied_date;
quit;

** Calculate number of days spent overseas **;
data overseas_time;
set overseas_spells;
if pos_ceased_date > "31DEC&year.:23:59:59.999"dt and pos_applied_date <
"01JUL&year.:00:00:00.000"dt
then time_to_add = 184;

else if pos_ceased_date > "31DEC&year.:23:59:59.999"dt and pos_applied_date
>= "01JUL&year.:00:00:00.000"dt
then time_to_add = ("31DEC&year.:23:59:59.999"dt - pos_applied_date) / 86400;

else if pos_ceased_date <= "31DEC&year.:23:59:59.999"dt and pos_applied_date
>= "01JUL&year.:00:00:00.000"dt
then time_to_add = (pos_ceased_date - pos_applied_date) / 86400;

else if pos_ceased_date <= "31DEC&year.:23:59:59.999"dt and pos_applied_date
< "01JUL&year.:00:00:00.000"dt
then time_to_add = (pos_ceased_date - "01JUL&year.:00:00:00.000"dt) / 86400;

if missing(pos_day_span_nbr) then time_to_add = 0;

run;

proc sql;
create table time_overseas_&year._2 as
select snz_uid, dob, dod, sex,
ROUND(SUM(time_to_add),1) as days_overseas
from overseas_time
group by snz_uid, dob, dod, sex;

create table time_overseas_&year._2 as
select *,
"01JUL&year."d as refdatestart format=date10.,
"31DEC&year."d as refdateend format=date10.
from time_overseas_&year._2;
quit;

%end;

DATA &thedata._OS_&start_yr._&end_yr.;
SET TIME_OVERSEAS:;
run;

PROC SORT DATA=&thedata._OS_&start_yr._&end_yr.;
by snz_uid refdatestart;
run;
```

```
DATA &thedata._OS_&start_yr._&end_yr.;
SET &thedata._OS_&start_yr._&end_yr.;

IF not missing(dod) and dod < refdatestart then delete;
run;


%mend os;

%os(totpop,2006, 2016);

************************************************************;
*************************************************************************
*;
** create days overseas or dead variable;
* note that in some cases days overseas or dead can add to more than the
number of days within the six month band,
(possibly because person died while overseas?)
- correcting for that in code below;
data os_tot_pop;
set totpop_os_2006_2016;
daysdead = refdateend - dod;
if daysdead<0 then daysdead=0;
days_os_or_dead = days_overseas + daysdead;
if days_os_or_dead>184 then days_os_or_dead=184;
if days_os_or_dead>181 & refdateend = '30JUN06'd then days_os_or_dead=181;
if days_os_or_dead>181 & refdateend = '30JUN07'd then days_os_or_dead=181;
if days_os_or_dead>182 & refdateend = '30JUN08'd then days_os_or_dead=182;
if days_os_or_dead>181 & refdateend = '30JUN09'd then days_os_or_dead=181;
if days_os_or_dead>181 & refdateend = '30JUN10'd then days_os_or_dead=181;
if days_os_or_dead>181 & refdateend = '30JUN11'd then days_os_or_dead=181;
if days_os_or_dead>182 & refdateend = '30JUN12'd then days_os_or_dead=182;
if days_os_or_dead>181 & refdateend = '30JUN13'd then days_os_or_dead=181;
if days_os_or_dead>181 & refdateend = '30JUN14'd then days_os_or_dead=181;
if days_os_or_dead>181 & refdateend = '30JUN15'd then days_os_or_dead=181;
if days_os_or_dead>182 & refdateend = '30JUN16'd then days_os_or_dead=182;
run;

*** DROP individuals who died prior to observation period;
data os_tot_pop2; set os_tot_pop;
if (dod ne .) and (dod < '01JUL2006'd) then delete;
run;

*drop extra date periods - limit to fiscal years;
data os_tot_pop3;
set os_tot_pop2;
if refdateend > '30JUN06'd ;
if refdateend < '31DEC16'd ;
run;


*transpose long to wide;
proc transpose data=os_tot_pop3 out=tot_pop_days_os_or_dead prefix=X;
by snz_uid;
id refdateend;
var days_os_or_dead;
run;
```

```
*give blanks max number of days within that period;
data tot_pop_days_os_or_dead_2;
set tot_pop_days_os_or_dead;
if X31DEC2006 = . THEN X31DEC2006 = 184;
if X30JUN2007 = . THEN X30JUN2007 = 181;
if X31DEC2007 = . THEN X31DEC2007 = 184;
if X30JUN2008 = . THEN X30JUN2008 = 182;
if X31DEC2008 = . THEN X31DEC2008 = 184;
if X30JUN2009 = . THEN X30JUN2009 = 181;
if X31DEC2009 = . THEN X31DEC2009 = 184;
if X30JUN2010 = . THEN X30JUN2010 = 181;
if X31DEC2010 = . THEN X31DEC2010 = 184;
if X30JUN2011 = . THEN X30JUN2011 = 181;
if X31DEC2011 = . THEN X31DEC2011 = 184;
if X30JUN2012 = . THEN X30JUN2012 = 182;
if X31DEC2012 = . THEN X31DEC2012 = 184;
if X30JUN2013 = . THEN X30JUN2013 = 181;
if X31DEC2013 = . THEN X31DEC2013 = 184;
if X30JUN2014 = . THEN X30JUN2014 = 181;
if X31DEC2014 = . THEN X31DEC2014 = 184;
if X30JUN2015 = . THEN X30JUN2015 = 181;
if X31DEC2015 = . THEN X31DEC2015 = 184;
if X30JUN2016 = . THEN X30JUN2016 = 182;
RUN;


* sum to get total number of days overseas and/or dead
* Create weight variable;
data tot_pop_days_os_or_dead_3;
set tot_pop_days_os_or_dead_2;
totdays_os_dead = sum(of _numeric_)-snz_uid;
wgt = (3653-totdays_os_dead)/3653; *this is a preliminary wgt variable - wgt2
created after sector use variables developped;
run;

*****************************************************************
NOTE: need to drop those who were overseas for the entire period;

** count total number of days overseas - drop those who were overseas more
than 3649 days;
proc sort data=os_tot_pop3;
by snz_uid refdatestart;
run;

data os_tot_pop4;
set os_tot_pop3;
spell + 1;
by snz_uid refdatestart;
if first.snz_uid then spell = 1;
label spell = 'overseas spell (half-year)';
run;

** Count TOTAL number of days overseas, total number of days dead & total
number of days overseas or dead;
data tot_pop_wide;
```

```
      do i = 1 to 20 until (last.snz_uid); /*1 to 20 as we have 20 6 month
bands */
            set os_tot_pop4;
            by snz_uid;

            retain
            days_overseas_Jul06Jun16 0;

            if first.snz_uid then do;
            days_overseas_Jul06Jun16 = 0;
      end;

      days_overseas_Jul06Jun16 = days_overseas_Jul06Jun16 + days_overseas;
end;

drop i spell days_overseas refdatestart refdateend daysdead days_os_or_dead;

label
days_overseas_Jul06Jun16 = 'total number of days overseas, 01/07/2006-
30/06/2016';
run;

** Drop individuals overseas for almost entire obs period and adjust if days
overseas/dead > 3650;
data tot_pop_wide_2;
set tot_pop_wide;
if days_overseas_Jul06Jun16 ge 3650 then delete;
run;


*********************************************************************************
*********;
* merge those who were not overseas for entire period into final cohort -
inner join;
proc sql;
create table tot_pop_wide_3 as
      select * from tot_pop_wide_2 a
      inner join (select snz_uid, totdays_os_dead, wgt from
tot_pop_days_os_or_dead_3) b
            on a.snz_uid = b.snz_uid;
quit;

* Create birth year and cohort variable;
data tot_pop_wide_4;
set tot_pop_wide_3;
birth_year = year(DOB);
if 1950 <= birth_year <= 1954 then cohort = 5054;
else if 1955 <= birth_year <= 1959 then cohort = 5559;
else if 1960 <= birth_year <= 1964 then cohort = 6064;
else if 1965 <= birth_year <= 1969 then cohort = 6569;
else if 1970 <= birth_year <= 1974 then cohort = 7074;
else if 1975 <= birth_year <= 1979 then cohort = 7579;
else if 1980 <= birth_year <= 1984 then cohort = 8084;
run;

* save file;
data steph.tot_pop_17Sep19;
```

```
set tot_pop_wide_4;
run;
```

```
/*************************************************************************
*************************************************************************
**********
AUTHOR: S. D'Souza (adapted from L. Richmond-Rakerd)
IDI REFRESH: idi_clean_20171020

PURPOSE: Calculate benefit use events and costs

DISCLAIMER:
The results in this [report, paper] are not official statistics.
They have been created for research purposes from the Integrated Data
Infrastructure (IDI), managed by Statistics New Zealand.
The opinions, findings, recommendations, and conclusions expressed in this
[report, paper etc] are those of the author(s), not Statistics NZ,
[Department X, or Organisation Y].
Access to the anonymised data used in this study was provided by Statistics
NZ under the security and confidentiality provisions of the Statistics Act
1975.
Only people authorised by the Statistics Act 1975 are allowed to see data
about a particular person, household, business, or organisation,
and the results in this [report, paper] have been confidentialised to protect
these groups from identification and to keep their data safe.
Careful consideration has been given to the privacy, security, and
confidentiality issues associated with using administrative and survey data
in the IDI.
Further detail can be found in the Privacy impact assessment for the
Integrated Data Infrastructure available from www.stats.govt.nz.

*************************************************************************
*************************************************************************
*********/

* specify libraries;
libname msd ODBC dsn=idi_clean_20171020_srvprd schema=msd_clean;
libname steph "/nas/DataLab/MAA/MAA2018-15 A small segment of the population
with large economic burden/Steph/Data";

** Format start & end dates;
* load data;
data benefits (rename=(msd_spel_spell_start_date=start_ben
msd_spel_spell_end_date=end_ben));
set msd.msd_spell;
run;

** Restrict observation period to between 01 July 2006 -- 30 June 2016;
* Missing end date indicates benefits ongoing;
data benefits2; set benefits;
if (end_ben = . and start_ben <= '30JUN2016'd)
or ('01JUL2006'd <= end_ben <= '30JUN2016'd)
or (end_ben > '30JUN2016'd and start_ben <= '30JUN2016'd);
run;

** Recode benefit code types char to numeric;
data benefits2; set benefits2;
benefit_type_num = msd_spel_servf_code*1;
run;
```

```
** Drop STUDENT UNEMPLOYMENT BENEFIT = 607;
data benefits3; set benefits2;
if benefit_type_num = 607 then delete;
run;

** Compute first & last dates for observation period;
data benefits4; set benefits3;
first_b = MDY (07, 01, 2006);
last_b = MDY (06, 30, 2016);
format first_b last_b date9.;

** Flag records as right-censored;
if (end_ben ne .) and (end_ben <= '30JUN2016'd) then right_censor = 0;
else if (end_ben = .) or (end_ben > '30JUN2016'd) then right_censor = 1;

** Flag records as left-censored;
if start_ben >= '01JUL2006'd then left_censor = 0;
else if start_ben < '01JUL2006'd then left_censor = 1;

* Assign right-censored cases last date in obs period
* Assign left-censored cases first date in obs period;
if right_censor = 1 then end_ben = last_b;
if left_censor = 1 then start_ben = first_b;

** Determine length of benefits between 01/07/2006 - 30/06/2016;
* Scaled to days;
benefit_length = (end_ben - start_ben);
run;

************************************************************;

** create benefit cost variables;

* 2 benefit rates: 2016 no children, 2016 with children. Rates obtained from
WINZ.
* SAS keeps crashing when I try to calculate all 3 benefit cost variables, so
doing them one at a time;

data benefits5; set benefits4;
      * change benefit length from days to weeks;
 ben_week = benefit_length/7;

* Do cost based on 2016 rates assuming no children (exception: sole parent
support)
  Based on rates from WINZ website;
if msd_spel_servf_code = '020' then bencost_2016nc = 218.86*ben_week; *
supported living payment;
if msd_spel_servf_code = '030' then bencost_2016nc = 218.86*ben_week; *
widow's benefit;
if msd_spel_servf_code = '115' then bencost_2016nc = 175.10*ben_week; *
unemployment benefit hardship;
if msd_spel_servf_code = '313' then bencost_2016nc = 175.10*ben_week; *
emergency maintenance allowance;
if msd_spel_servf_code = '320' then bencost_2016nc = 218.86*ben_week; *
invalids benefit;
if msd_spel_servf_code = '330' then bencost_2016nc = 218.86*ben_week; *
widow's benefit;
```

```
if msd_spel_servf_code = '365' then bencost_2016nc = 325.98*ben_week; * sole
parent support;
if msd_spel_servf_code = '366' then bencost_2016nc = 218.86*ben_week; * DPB
woman alone;
if msd_spel_servf_code = '367' then bencost_2016nc = 218.86*ben_week; * DPB
caring for sick or infirm;
if msd_spel_servf_code = '370' then bencost_2016nc = 218.86*ben_week; *
supported living payment;
if msd_spel_servf_code = '600' then bencost_2016nc = 175.10*ben_week; *
sickness benefit;
if msd_spel_servf_code = '603' then bencost_2016nc = 175.10*ben_week; *
youth/young parent payment;
if msd_spel_servf_code = '608' then bencost_2016nc = 175.10*ben_week; *
unemployment benefit training;
if msd_spel_servf_code = '610' then bencost_2016nc = 175.10*ben_week; *
unemployment benefit;
if msd_spel_servf_code = '611' then bencost_2016nc = 175.10*ben_week; *
emergency benefit;
if msd_spel_servf_code = '665' then bencost_2016nc = 325.98*ben_week; * sole
parent support;
if msd_spel_servf_code = '666' then bencost_2016nc = 218.86*ben_week; * DPB
woman alone;
if msd_spel_servf_code = '675' then bencost_2016nc = 175.10*ben_week; *
jobseeker;

* Do cost based on 2016 rates assuming they have children (exception: Widow's
benefit/DPB woman alone)
  Based on rates from WINZ website;
if msd_spel_servf_code = '020' then bencost_2016c = 231.36*ben_week; *
supported living payment;
if msd_spel_servf_code = '030' then bencost_2016c = 218.86*ben_week; *
widow's benefit;
if msd_spel_servf_code = '115' then bencost_2016c = 187.60*ben_week; *
unemployment benefit hardship;
if msd_spel_servf_code = '313' then bencost_2016c = 325.98*ben_week; *
emergency maintenance allowance;
if msd_spel_servf_code = '320' then bencost_2016c = 231.36*ben_week; *
invalids benefit;
if msd_spel_servf_code = '330' then bencost_2016c = 218.86*ben_week; *
widow's benefit;
if msd_spel_servf_code = '365' then bencost_2016c = 325.98*ben_week; * sole
parent support;
if msd_spel_servf_code = '366' then bencost_2016c = 218.86*ben_week; * DPB
woman alone;
if msd_spel_servf_code = '367' then bencost_2016c = 231.36*ben_week; * DPB
caring for sick or infirm;
if msd_spel_servf_code = '370' then bencost_2016c = 231.36*ben_week; *
supported living payment;
if msd_spel_servf_code = '600' then bencost_2016c = 187.60*ben_week; *
sickness benefit;
if msd_spel_servf_code = '603' then bencost_2016c = 187.60*ben_week; *
youth/young parent payment;
if msd_spel_servf_code = '608' then bencost_2016c = 187.60*ben_week; *
unemployment benefit training;
if msd_spel_servf_code = '610' then bencost_2016c = 187.60*ben_week; *
unemployment benefit;
```

```
if msd_spel_servf_code = '611' then bencost_2016c = 325.98*ben_week; *
emergency benefit;
if msd_spel_servf_code = '665' then bencost_2016c = 325.98*ben_week; * sole
parent support;
if msd_spel_servf_code = '666' then bencost_2016c = 218.86*ben_week; * DPB
woman alone;
if msd_spel_servf_code = '675' then bencost_2016c = 187.60*ben_week; *
jobseeker;

run;

* get number of benefit spells;
proc sort data=benefits5;
      by snz_uid start_ben;
data benefits6;
      set benefits5;
      ben_num + 1;
      by snz_uid start_ben;
      if first.snz_uid then ben_num = 1;
run;

* find out max number of benefit spells;
proc means data=benefits6;
var ben_num;
run;

*enumerate;
data benefits7;
      do i = 1 to 250 until (last.snz_uid); /* arbitrary number selected that
is greater than the max number of spells */
            set benefits6;
            by snz_uid;
            retain      totcost_ben2016c 0      totcost_ben2016nc 0
      blength_Jul06Jun16 0;
            if first.snz_uid then do;
                        totcost_ben2016c = 0;   totcost_ben2016nc = 0;
      blength_Jul06Jun16  = 0;
            end;

totcost_ben2016c = totcost_ben2016c + bencost_2016c;
totcost_ben2016nc = totcost_ben2016nc + bencost_2016nc;
blength_Jul06Jun16  = blength_Jul06Jun16 + benefit_length;
anyben_Jul06Jun16 = 1;

format totcost_ben2016c totcost_ben2016nc dollar10.2;

end;

drop i;
run;

*truncate benefit length to max. number of days within exposure period - as
there are some overlapping spells;
data benefits8;
set benefits7;
if blength_Jul06Jun16 > 3653 then trunc_blength = 3653;
else trunc_blength = blength_Jul06Jun16;
```

```
run;

** save data;
* keep only necessary variables. Keeping only benefit costs assuming
dependents;
data steph.benefits_07Sep19;
set benefits8;
keep snz_uid anyben_Jul06Jun16 trunc_blength blength_Jul06Jun16
totcost_ben2016c totcost_ben2016nc;
run;
************************************;
```

```
/*************************************************************************
*************************************************************************
**********
AUTHOR: S. D'Souza (adapted from L. Richmond-Rakerd)
IDI REFRESH: idi_clean_20171020

PURPOSE: Calculate hosp use events and costs

DISCLAIMER:
The results in this [report, paper] are not official statistics.
They have been created for research purposes from the Integrated Data
Infrastructure (IDI), managed by Statistics New Zealand.
The opinions, findings, recommendations, and conclusions expressed in this
[report, paper etc] are those of the author(s), not Statistics NZ,
[Department X, or Organisation Y].
Access to the anonymised data used in this study was provided by Statistics
NZ under the security and confidentiality provisions of the Statistics Act
1975.
Only people authorised by the Statistics Act 1975 are allowed to see data
about a particular person, household, business, or organisation,
and the results in this [report, paper] have been confidentialised to protect
these groups from identification and to keep their data safe.
Careful consideration has been given to the privacy, security, and
confidentiality issues associated with using administrative and survey data
in the IDI.
Further detail can be found in the Privacy impact assessment for the
Integrated Data Infrastructure available from www.stats.govt.nz.

*************************************************************************
*************************************************************************
*********/

* libraries;
libname moh ODBC dsn=idi_clean_20171020_srvprd schema=moh_clean;
libname steph "/nas/DataLab/MAA/MAA2018-15 A small segment of the population
with large economic burden/Steph/Data";


* apply filtering criteria specific to hosp diagnosis dataset
if filter requires info from both diag and event datafiles, then filter has
'pre';
data hosp_diag;
set moh.pub_fund_hosp_discharges_diag;
if moh_dia_diag_sequence_code  = '10' and substr(moh_dia_clinical_code,1,3) =
'Z38' then moh_wellbaby = '1';
else moh_wellbaby = '0';

if moh_dia_diag_sequence_code in ('10','20') and moh_dia_clinical_code in
('Z510','Z511','Z512') then pre_moh_sdchemo = '1'; *event + diag;

if moh_dia_diag_sequence_code  = '10' and moh_dia_clinical_code in
('Z742','Z755') then pre_moh_dsscase = '1'; * event + diag;
if moh_dia_op_date = '' then op_null = '1';

if moh_dia_diag_sequence_code  = '10' and moh_dia_clinical_code in
('Z763','Z764') then pre_moh_nontreated = '1'; *diag;
```

```
if moh_dia_diag_sequence_code in ('10','20','30','40','50','60') and
moh_dia_clinical_code in ('Z530','Z531','Z532','Z538','Z539') then
pre_moh_nontreated = '2'; * event + diag;

if moh_dia_diag_sequence_code  = '10' and moh_dia_clinical_code = 'Z513' then
pre_moh_bloodtrans = '1'; * event + diag;
run;


** Connect to hosp event dataset;
data hosp_event (rename=(moh_evt_evst_date=hosp_start
moh_evt_even_date=hosp_end));
set moh.pub_fund_hosp_discharges_event;
run;


** Restrict observation period to between 01 July 2006 -- 30 June 2016
* NO spells coded as missing at start or end of data capture (all assigned
start and end dates).
* Include spells that started & ended on same day;
data hosp_event2; set hosp_event;
if (hosp_start <= '01JUL2006'd and hosp_end >= '01JUL2006'd)
or ('01JUL2006'd <= hosp_end <= '30JUN2016'd)
or (hosp_end >= '30JUN2016'd and hosp_start <= '30JUN2016'd);
run;

** Censoring;
data hosp_event3; set hosp_event2;
first_h = MDY (07, 01, 2006);
last_h = MDY (06, 30, 2016);
format first_h last_h date9.;

* Flag records as right-censored;
if hosp_end <= '30JUN2016'd then right_censor = 0;
else if hosp_end > '30JUN2016'd then right_censor = 1;

* Flag records as left-censored;
if hosp_start >= '01JUL2006'd then left_censor = 0;
else if hosp_start < '01JUL2006'd then left_censor = 1;

* Assign right-censored cases last date in hosp file
* Assign left-censored cases first date in hosp file;
if right_censor = 1 then hosp_end = last_h;
if left_censor = 1 then hosp_start = first_h;

** Compute different LoS vars (scaled to days);
hosp_length = (hosp_end - hosp_start);          * same-day cases assigned 0;
run;

** merge in diag filter info;
proc sql;
      create table hosp as
            select * from hosp_event3 a
            left join hosp_diag b
                  on a.moh_evt_event_id_nbr = b.moh_dia_event_id_nbr;
quit;
```

```
*Setting up MoH filtering criteria;
data hosp2;
set hosp;
if moh_wellbaby ne '1' then moh_wellbaby = '0';

if moh_evt_drg_31_code in ('951','952','955','956') then moh_errordrg = '1';
else moh_errordrg = '0';

if moh_evt_drg_current_text in ('572','L61Y','L61Z') then moh_renaldialysis =
'1';
else moh_renaldialysis = '0';

if moh_evt_los_nbr = 0 and moh_evt_end_type_code ne 'DD' and
moh_evt_hlth_spec_code in ('M05','M06','M07','M08') then moh_aedaystay = '1';
*event;
else moh_aedaystay = '0';

if moh_evt_drg_31_code = '940' then rehab = '1'; *event;
else rehab = '0';

if moh_evt_los_nbr = 0 and pre_moh_sdchemo = '1' then moh_sdchemo = '1';
*event + diag;
else moh_sdchemo = '0';

if moh_evt_los_nbr < 2 and moh_evt_drg_current_text in ('174','175','E63Z')
then moh_sleepapnoea = '1'; *event;
else moh_sleepapnoea = '0';

if moh_evt_dom_cd_code = '9999' then moh_overseas = '1'; *check freqs before
using as filter; * event;
else moh_overseas = '0';

if substr(moh_evt_hlth_spec_code,1,1) = 'D' then moh_dsscase = '1';
*disability support service;
else if moh_evt_drg_current_text in ('940','941','Z60A','Z60B','Z60C') then
moh_dsscase = '1';
else if pre_moh_dsscase = '1' then moh_dsscase = '1'; * event + diag;
else if moh_evt_facility_code in
('3217','3220','3232','3235','3237','3238','3614','3912','3913','4015','4017'
,'4024',
'4031','4222','5750','5814','5229','5330','3226','3228','4314','5914') and
moh_evt_hlth_spec_code not in ('P10',
'P11','P12','P13','P14','P15','P16','P17','P18') and op_null = '1' and
moh_evt_los_nbr > 10 then moh_dsscase = '1';
else moh_dsscase = '0';

if pre_moh_nontreated = '1' then moh_nontreated = '1'; *diag;
else if op_null = '1' and moh_evt_adm_type_code not in ('AC','ZC') and
moh_evt_los_nbr < 2 and
pre_moh_nontreated = '2'
then moh_nontreated = '1'; * event + diag;
else moh_nontreated = '0';

if moh_evt_los_nbr = 0 and moh_evt_adm_type_code not in ('AC','ZC') and
pre_moh_bloodtrans = '1'
then moh_bloodtrans = '1'; * event + diag;
```

```
else moh_bloodtrans = '0'; * can do based on diag1;
run;

*check if drg 7 codes that are different from prev drg versions are in
dataset;
proc freq data=hosp2;
table moh_evt_drg_current_text;
where moh_evt_drg_current_text = 'Z60Z';
run;
* not there in dataset;

* create 'filter' variable;
data hosp3;
set hosp2;
if moh_wellbaby = '1' then moh_filter = '1';
if moh_errordrg = '1' then moh_filter = '1';
if moh_renaldialysis = '1' then moh_filter = '1';
if rehab = '1' then moh_filter = '1';
if moh_sdchemo = '1' then moh_filter = '1';
if moh_sleepapnoea = '1' then moh_filter = '1';
if moh_dsscase = '1' then moh_filter = '1';
if moh_nontreated = '1' then moh_filter = '1';
if moh_bloodtrans = '1' then moh_filter = '1';
if moh_overseas = '1' then moh_filter = '1';
if moh_aedaystay = '1' then moh_filter = '1';
else moh_filter = '0';
run;

* get event IDs for filters;
proc freq data=hosp3 noprint;
table moh_evt_event_id_nbr / out=event_filter;
where moh_filter='1';
run;

*create filter variable;
data event_filter2;
set event_filter;
moh_filter_event = 1;
drop COUNT percent;
run;

* merge back in with most recent hospital events file (hosp_event3);
proc sql;
      create table hosp4 as
            select * from hosp_event3 a
            left join event_filter2 b
                  on a.moh_evt_event_id_nbr = b.moh_evt_event_id_nbr;
quit;

* how many events need to be filtered out;
proc freq data=hosp4;
table moh_filter_event;
run;

* remove filters;
data hosp5;
set hosp4;
```

```
where moh_filter_event ne 1;
run;

*******************************************;
** get diag data again - to adjust for birth events;
* restrict to ICD-10;
data diag;
set moh.pub_fund_hosp_discharges_diag;
if moh_dia_clinical_sys_code  in ('10','11','12','13','14') ;
run;

* restict to pregnancy delivery-related event;
data diag2;
set diag;
if moh_dia_clinical_code in
('O60','O600','O601','O602','O603','O610','O611','O618','O619','O620','O621',
'O622','O623','O624','O628','O629',
'O630','O631','O632','O639','O640','O641','O642','O643','O644','O645','O648',
'O649','O650','O651','O652','O653',
'O654','O655','O658','O659','O660','O661','O662','O663','O664','O665','O668',
'O669','O670','O678','O679','O680',
'O681','O682','O683','O688','O689','O690','O691','O692','O693','O694','O695',
'O698','O699','O700','O701','O702',
'O703','O709','O710','O7100','O7101','O7102','O711','O7110','O7111','O712','O
713','O714','O715','O716','O717','O718',
'O7181','O7182','O7188','O719','O720','O721','O722','O723','O730','O731','O74
1','O742','O743','O744','O745','O746',
'O747','O748','O750','O751','O752','O753','O754','O755','O756','O757','O758',
'O759','O80','O81','O82','O83',
'O840','O841','O842','O8481','O8482','Z370');
run;

* get event ids associated with birth events;
proc freq data=diag2 noprint;
table moh_dia_event_id_nbr /out=diag3;
run;

* create birth event flag;
data birthevents (keep=moh_evt_event_id_nbr birthevent);
set diag3;
moh_evt_event_id_nbr = moh_dia_event_id_nbr;
birthevent=1;
run;

* merge;
proc sql;
      create table hosp_births as
            select * from hosp5 a
            left join birthevents b
                  on a.moh_evt_event_id_nbr = b.moh_evt_event_id_nbr;
quit;

*look at hospital length of stay for birt events;
proc univariate data=hosp_births;
var hosp_length;
where birthevent = 1;
run;
```

```
* adjust for birth event-related length of stay - 90th percentile value used
for adjustment;
data hosp_births2;
set hosp_births;
hosp_length_nobt = hosp_length;
if birthevent=1 then hosp_length_nobt = hosp_length-5;
if hosp_length_nobt<0 then hosp_length_nobt=0;
run;

proc means data=hosp_births2;
var hosp_length hosp_length_nobt;
run;
* means etc don't change by much;

** HOSPITAL ADMISSIONS;
* Enumerate admissions, LoS, costweights, etc.
* Count total admissions;

** use final fiscal year (2015/16) multiplier to calculate costs;
data hosp_births3; set hosp_births2;
PUcost_Jul15Jun16 = moh_evt_cost_weight_amt*4751.58;
format PUcost_Jul15Jun16 dollar10.2;
run;

** HOSPITAL ADMISSIONS;
* Enumerate admissions, LoS, costweights, etc.
* Count total admissions;
proc sort data=hosp_births3;
by snz_uid hosp_start;
data hosp_births4; set hosp_births3;
adm_num_btadj + 1;
by snz_uid hosp_start;
if first.snz_uid then adm_num_btadj = 1;
run;

* find out max number of hospitalisations spells;
proc means data=hosp_births4;
var adm_num_btadj;
run;

*convert to wide format;
data hospital_flat_ALL;
      do i = 1 to 1200 until (last.snz_uid); /* arbitrary number greater than
the max # of admissions */
            set hosp_births4;
            by snz_uid;
            retain      totlos_BTadj      0     totcost_BTadjmult  0
      original_los      0;
            if first.snz_uid then do;
                        totlos_BTadj = 0; totcost_BTadjmult = 0; original_los
= 0;
            end;

      totlos_BTadj = totlos_BTadj + hosp_length_nobt; * length of stay
adjusted for birth events;
```

```
      original_los = original_los + hosp_length;            * original length
of stay;
      totcost_BTadjmult = totcost_BTadjmult + PUcost_Jul15Jun16; * cost with
final fiscal year multiplier;
      end;

      format totcost_BTadjmult dollar10.2;

      keep snz_uid totlos_BTadj original_los totcost_BTadjmult
totcost_BTadjmult;
      run;

* create hospitalisation flag - for both original los and birth-adjusted los;
data hospital_flat_ALL2;
set hospital_flat_ALL;
if original_los > 0 then anyhosp_overnight = 1;
else anyhosp_overnight = 0;
if totlos_BTadj > 0 then anyhospBTadj_overnight = 1;
else anyhospBTadj_overnight = 0;
run;

*save dataset;
data steph.hospitalisation_20Sep19; set hospital_flat_ALL2;
run;
```

```
/*************************************************************************
*************************************************************************
**********
AUTHOR: S. D'Souza
IDI REFRESH: idi_clean_20171020

PURPOSE: Calculate Pharms use and costs

DISCLAIMER:
The results in this [report, paper] are not official statistics.
They have been created for research purposes from the Integrated Data
Infrastructure (IDI), managed by Statistics New Zealand.
The opinions, findings, recommendations, and conclusions expressed in this
[report, paper etc] are those of the author(s), not Statistics NZ,
[Department X, or Organisation Y].
Access to the anonymised data used in this study was provided by Statistics
NZ under the security and confidentiality provisions of the Statistics Act
1975.
Only people authorised by the Statistics Act 1975 are allowed to see data
about a particular person, household, business, or organisation,
and the results in this [report, paper] have been confidentialised to protect
these groups from identification and to keep their data safe.
Careful consideration has been given to the privacy, security, and
confidentiality issues associated with using administrative and survey data
in the IDI.
Further detail can be found in the Privacy impact assessment for the
Integrated Data Infrastructure available from www.stats.govt.nz.

*************************************************************************
*************************************************************************
*********/

* libraries;
libname moh ODBC dsn=idi_clean_20171020_srvprd schema=moh_clean;
libname metadata ODBC dsn=idi_metadata_srvprd
schema=clean_read_CLASSIFICATIONS;
libname steph "/nas/DataLab/MAA/MAA2018-15 A small segment of the population
with large economic burden/Steph/Data";

************************************************************************'

** NOTE: Because the pharms data is so large - we initially loaded data by
each fiscal year and data processing was done using that data.
      This is not efficient and very time consuming. The code below is more
efficient.
      Specifically, to save on processing power and speed, we restrict to
snz_uids within our total pop and load pharms data only for those
individuals;

** Pharms filtering - as per MoH guidelines;
* load relevant snz_uids and dod (need dod for filtering) from total pop and
left join in necessary pharms info;
proc sql;
      create table pharms as
            select a.snz_uid, a.dod, b.moh_pha_birth_year_nbr,
b.moh_pha_dispensed_date, b.moh_pha_dim_form_pack_code,
```

```
                        b.moh_pha_order_type_code,
b.moh_pha_patient_contrib_exc_gst_, b.moh_pha_remimburs_cost_exc_gst_a
                              from steph.tot_pop_17Sep19 a
                              left join moh.pharmaceutical b
                                    on a.snz_uid = b.snz_uid
                                          where '01JUL2006'd <=
b.moh_pha_dispensed_date <= '30JUN2016'd;
quit;

* create death flag;
data pharms_2;
set pharms (rename=(moh_pha_dispensed_date = disp_date));
if dod ne . and dod < disp_date then death_flag = 1;
else death_flag = 0;
run;

* drop rows where death_flag = 1;
data pharms_3;
set pharms_2;
where death_flag = 0;
dim_form_pack_subsidy_key = input(moh_pha_dim_form_pack_code,8.); * convert
to numeric and allow for merging;
run;

* merge in chemical ids, etc.;
proc sql;
create table pharms_4 as
      select a.*, b.dim_form_pack_subsidy_key, b.chemical_id,
b.formulation_id
      from pharms_3 a
            left join metadata.moh_dim_form_pack_subsidy_code b
                  on a.dim_form_pack_subsidy_key =
b.dim_form_pack_subsidy_key
      ;
quit;

* exclude null formulation ids;
data pharms_5;
set pharms_4;
where formulation_id ne . and formulation_id ne 391725;
run;

** Restrict to order type 1 (prescription) or order type 7 (oncology);
data pharms_6;
set pharms_5;
where moh_pha_order_type_code in ('1','7');
run;

*****************************************************************;
* Enumerate;

*Enumerate number of scripts;
proc sort data=pharms_6;
by snz_uid disp_date;
data pharms_7;
set pharms_6;
script_num + 1;
```

```
by snz_uid disp_date;
if first.snz_uid then script_num = 1;
run;

* Find out max number of scripts;
proc means data=pharms_7;
var script_num;
run;

*calculate total cost & convert to wide format;
data pharms_wide;
      do i = 1 to 34500 until (last.snz_uid); /* arbitrary number selected
that is greater than the max number of claims */
            set pharms_7;
            by snz_uid;
            retain      tot_num_scripts  0      totcost_patient  0
      totcost_reimburs    0;
            if first.snz_uid then do;
                        tot_num_scripts = 0; totcost_patient = 0;
totcost_reimburs    = 0;
            end;

tot_num_scripts = tot_num_scripts + 1;
totcost_patient = totcost_patient + moh_pha_patient_contrib_exc_gst_;
totcost_reimburs = totcost_reimburs + moh_pha_remimburs_cost_exc_gst_a;
anyscript_Jul06Jun16 = 1; /* All individuals in dataset should be coded as
'1' for any claim*/

format totcost_patient totcost_reimburs dollar10.2;

end;
drop i script_num moh_pha_patient_contrib_exc_gst_
moh_pha_remimburs_cost_exc_gst_a;
run;

* create final cost variable and save;
data steph.pharms_24Sep19;
set pharms_wide;

pharms_cost = totcost_patient + totcost_reimburs;
format pharms_cost dollar10.2;

keep snz_uid tot_num_scripts pharms_cost anyscript_Jul06Jun16;
run;
```

```
/*************************************************************************
*************************************************************************
**********
AUTHOR: S. D'Souza
IDI REFRESH: idi_clean_20171020

PURPOSE: Calculate ACC use and costs

DISCLAIMER:
The results in this [report, paper] are not official statistics.
They have been created for research purposes from the Integrated Data
Infrastructure (IDI), managed by Statistics New Zealand.
The opinions, findings, recommendations, and conclusions expressed in this
[report, paper etc] are those of the author(s), not Statistics NZ,
[Department X, or Organisation Y].
Access to the anonymised data used in this study was provided by Statistics
NZ under the security and confidentiality provisions of the Statistics Act
1975.
Only people authorised by the Statistics Act 1975 are allowed to see data
about a particular person, household, business, or organisation,
and the results in this [report, paper] have been confidentialised to protect
these groups from identification and to keep their data safe.
Careful consideration has been given to the privacy, security, and
confidentiality issues associated with using administrative and survey data
in the IDI.
Further detail can be found in the Privacy impact assessment for the
Integrated Data Infrastructure available from www.stats.govt.nz.

*************************************************************************
*************************************************************************
*********/

* libraries;
libname acc ODBC dsn=idi_clean_20171020_srvprd schema=acc_clean;
libname steph "/nas/DataLab/MAA/MAA2018-15 A small segment of the population
with large economic burden/Steph/Data";

**load acc data;
* restrict to time period of interest and only accepted claims;
proc sql;
create table acc as
      select * from acc.claims
      where ('01JUL2006'd <= acc_cla_lodgement_date <= '30JUN2016'd) and
acc_cla_decision_text = 'ACCEPT';
quit;

*Enumerate number of acc claims;
proc sort data=acc;
by snz_uid acc_cla_lodgement_date;
data acc2;
set acc;
claim_num + 1;
by snz_uid acc_cla_lodgement_date;
if first.snz_uid then claim_num = 1;
run;

* Find out max number of acc claims;
```

```
proc means data=acc2;
var claim_num;
run;

*calculate total cost & convert to wide format;
data acc_wide;
      do i = 1 to 170 until (last.snz_uid); /* arbitrary number selected that
is greater than the max number of claims */
            set acc2;
            by snz_uid;
            retain      tot_num_claims  0 totclaimcost_Jul06Jun16 0;
            if first.snz_uid then do;
                        tot_num_claims = 0; totclaimcost_Jul06Jun16 = 0;
            end;

tot_num_claims = tot_num_claims + 1;
totclaimcost_Jul06Jun16 = totclaimcost_Jul06Jun16 +
acc_cla_claim_costs_to_date_ex_g;
anyclaim_Jul06Jun16 = 1; /* All individuals in dataset should be coded as '1'
for any claim*/

format totclaimcost_Jul06Jun16 dollar10.2;

end;
drop i claim_num;
run;

*save final datafile;
data steph.acc_20Sep19;
set acc_wide;
keep snz_uid tot_num_claims totclaimcost_Jul06Jun16 anyclaim_Jul06Jun16;
run;
```

```
/*************************************************************************
*************************************************************************
**********
AUTHOR: S. D'Souza (adapted from B. Milne)
IDI REFRESH: idi_clean_20171020

PURPOSE: Calculate convictions

DISCLAIMER:
The results in this [report, paper] are not official statistics.
They have been created for research purposes from the Integrated Data
Infrastructure (IDI), managed by Statistics New Zealand.
The opinions, findings, recommendations, and conclusions expressed in this
[report, paper etc] are those of the author(s), not Statistics NZ,
[Department X, or Organisation Y].
Access to the anonymised data used in this study was provided by Statistics
NZ under the security and confidentiality provisions of the Statistics Act
1975.
Only people authorised by the Statistics Act 1975 are allowed to see data
about a particular person, household, business, or organisation,
and the results in this [report, paper] have been confidentialised to protect
these groups from identification and to keep their data safe.
Careful consideration has been given to the privacy, security, and
confidentiality issues associated with using administrative and survey data
in the IDI.
Further detail can be found in the Privacy impact assessment for the
Integrated Data Infrastructure available from www.stats.govt.nz.

*************************************************************************
*************************************************************************
*********/

* libraries;
libname moj ODBC dsn=idi_clean_20171020_srvprd schema=moj_clean;
libname steph "/nas/DataLab/MAA/MAA2018-15 A small segment of the population
with large economic burden/Steph/Data";

** Connect to moj data;
* using charge outcome date;
data charges (rename=(moj_chg_charge_outcome_date=outcome_date));
set moj.charges;
where ('01JUL2006'd <= moj_chg_charge_outcome_date <= '30JUN2016'd);
run;

*restricting to convictions;
data convictions; set charges;
if moj_chg_charge_outcome_type_code in
('CNV','CNVD','CNVS','COAD','DCP','J39J','MIPS34',

      'COND','CONV','J118','CCMD','COCM','CVOC');
run;


*Enumerate number of convictions - including traffic offences;
proc sort data=convictions;
by snz_uid outcome_date;
data conv;
```

```
set convictions;
conv_num + 1;
by snz_uid outcome_date;
if first.snz_uid then conv_num = 1;
run;

* Find out max number of convictions;
proc means data=conv;
var conv_num;
run;

*calculate total conv & convert to wide format;
data conv_wide;
      do i = 1 to 450 until (last.snz_uid); /* arbitrary number selected that
is greater than the max number of convictions */
            set conv;
            by snz_uid;
                  retain      tot_num_conv  0;
                  if first.snz_uid then do;
                        tot_num_conv = 0;
            end;

tot_num_conv = tot_num_conv + 1;
anyconv_Jul06Jun16 = 1; /* All individuals in dataset should be coded as '1'
for any conviction*/

end;
drop i conv_num;
run;

*****************************************************************;
** Do the same but excluding traffic offences

*****remove traffic offences;
data convictions_nt; set convictions;
if substr(moj_chg_offence_code,1,1) ne "8";
if substr(moj_chg_offence_code,1,1) ne "C";
if substr(moj_chg_offence_code,1,1) ne "D";
if substr(moj_chg_offence_code,1,1) ne "E";
if substr(moj_chg_offence_code,1,1) ne "F";
if substr(moj_chg_offence_code,1,1) ne "G";
if substr(moj_chg_offence_code,1,1) ne "H";
if substr(moj_chg_offence_code,1,1) ne "J";
if substr(moj_chg_offence_code,1,1) ne "K";
if substr(moj_chg_offence_code,1,1) ne "L";
if substr(moj_chg_offence_code,1,1) ne "M";
if substr(moj_chg_offence_code,1,1) ne "N";
if substr(moj_chg_offence_code,1,1) ne "O";
if substr(moj_chg_offence_code,1,1) ne "P";
if substr(moj_chg_offence_code,1,1) ne "R";
if substr(moj_chg_offence_code,1,1) ne "S";
if substr(moj_chg_offence_code,1,1) ne "V";
if substr(moj_chg_offence_code,1,3) ne "B15";
if substr(moj_chg_offence_code,1,3) ne "B16";
if substr(moj_chg_offence_code,1,3) ne "B17";
if moj_chg_offence_code ne "B181";
if moj_chg_offence_code ne "B182";
```

```
run;

*Enumerate excluding traffic offences;
proc sort data=convictions_nt;
by snz_uid outcome_date;
data conv_nt;
set convictions_nt;
nt_conv_num + 1;
by snz_uid outcome_date;
if first.snz_uid then nt_conv_num = 1;
run;

* Find out max number of non-traffic convictions;
proc means data=conv_nt;
var nt_conv_num;
run;

*calculate total conv & convert to wide format;
data conv_wide_nt;
      do i = 1 to 450 until (last.snz_uid); /* arbitrary number selected that
is greater than the max number of convictions */
            set conv_nt;
            by snz_uid;
                  retain      tot_num_conv_nt  0;
                  if first.snz_uid then do;
                        tot_num_conv_nt = 0;
            end;

tot_num_conv_nt = tot_num_conv_nt + 1;
anyconv_nt_Jul06Jun16 = 1; /* All individuals in dataset should be coded as
'1' for any no-traffic conviction*/

end;
drop i nt_conv_num;
run;

* merge datasets to get both use events (including and excluding traffic
convictions) in one datafile;
proc sql;
      create table conv_all as
            select a.snz_uid, a.tot_num_conv, a.anyconv_Jul06Jun16,
b.tot_num_conv_nt, b.anyconv_nt_Jul06Jun16
                  from conv_wide a
                  left join conv_wide_nt b
                        on a.snz_uid = b.snz_uid;
quit;


*save final datafile;
data steph.conv_23Sep19;
set conv_all;
run;
```

```
/*************************************************************************
*************************************************************************
**********
AUTHOR: S. D'Souza
IDI REFRESH: idi_clean_20171020

PURPOSE: Merge with total population, get data into final shape, get
wgt2 variable (this accounts for whether or not individual is in the
resident population)

DISCLAIMER:
The results in this [report, paper] are not official statistics.
They have been created for research purposes from the Integrated Data
Infrastructure (IDI), managed by Statistics New Zealand.
The opinions, findings, recommendations, and conclusions expressed in this
[report, paper etc] are those of the author(s), not Statistics NZ,
[Department X, or Organisation Y].
Access to the anonymised data used in this study was provided by Statistics
NZ under the security and confidentiality provisions of the Statistics Act
1975.
Only people authorised by the Statistics Act 1975 are allowed to see data
about a particular person, household, business, or organisation,
and the results in this [report, paper] have been confidentialised to protect
these groups from identification and to keep their data safe.
Careful consideration has been given to the privacy, security, and
confidentiality issues associated with using administrative and survey data
in the IDI.
Further detail can be found in the Privacy impact assessment for the
Integrated Data Infrastructure available from www.stats.govt.nz.

*************************************************************************
*************************************************************************
*********/

*libraries;
libname steph "/nas/DataLab/MAA/MAA2018-15 A small segment of the population
with large economic burden/Steph/Data";
libname data ODBC dsn=idi_clean_20171020_srvprd schema=data;
libname census odbc dsn=idi_clean_20171020_srvprd schema=cen_clean;

* merge sector data with total population;
proc sql;
      create table tot_pop as
            select * from steph.tot_pop_17Sep19 a
                  left join steph.benefits_07Sep19 b on a.snz_uid = b.snz_uid
                  left join steph.hospitalisation_20Sep19 c on a.snz_uid =
c.snz_uid
                  left join steph.acc_20Sep19 d on a.snz_uid = d.snz_uid
                  left join steph.conv_23Sep19 e on a.snz_uid = e.snz_uid
                  left join steph.pharms_24Sep19 f on a.snz_uid = f.snz_uid;
quit;

* create 0s for flags and assign value of 0 for those with no activities in
sector;
data tot_pop_2; set tot_pop;

* benefits;
```

```
if anyben_Jul06Jun16 ne 1 then anyben_Jul06Jun16 = 0;
if blength_Jul06Jun16 = . then blength_Jul06Jun16 = 0;
if trunc_blength = . then trunc_blength = 0;
if totcost_ben2016c = . then totcost_ben2016c = 0;
if totcost_ben2016nc = . then totcost_ben2016nc = 0;

* hospitalisations;
if anyhosp_overnight ne 1 then anyhosp_overnight = 0;
if anyhospBTadj_overnight ne 1 then anyhospBTadj_overnight = 0;
if original_los = . then original_los = 0;
if totlos_BTadj = . then totlos_BTadj = 0;
if totcost_BTadjmult = . then totcost_BTadjmult = 0;

* acc;
if anyclaim_Jul06Jun16 ne 1 then anyclaim_Jul06Jun16 = 0;
if tot_num_claims = . then tot_num_claims = 0;
if totclaimcost_Jul06Jun16 = . then totclaimcost_Jul06Jun16 = 0;

* convictions;
if anyconv_Jul06Jun16 ne 1 then anyconv_Jul06Jun16 = 0;
if anyconv_nt_Jul06Jun16 ne 1 then anyconv_nt_Jul06Jun16 = 0;
if tot_num_conv = . then tot_num_conv = 0;
if tot_num_conv_nt = . then tot_num_conv_nt = 0;

* pharmaceuticals;
if anyscript_Jul06Jun16 ne 1 then anyscript_Jul06Jun16 = 0;
if tot_num_scripts = . then tot_num_scripts = 0;
if pharms_cost = . then pharms_cost = 0;
run;

** create wgt2 - accounts for those not in resident population;
* load resident pop data;
data res_pop;
set data.snz_res_pop;
run;

* remove duplicate snz_uids;
proc freq data=res_pop noprint;
table snz_uid /out=res_pop_ids;
run;

* create flag for those in resident pop;
data res_pop_ids2;
set res_pop_ids;
drop count percent;
res_pop = 1;
run;

* merge resident pop flag in with total pop;
proc sql;
      create table res_tot_pop as
            select * from tot_pop_2 a
            left join res_pop_ids2 b
                  on a.snz_uid = b.snz_uid;
quit;

*identify those not in resident population;
```

```
data res_tot_pop2; set res_tot_pop;
if res_pop ne 1 then res_pop = 0;
run;

proc freq data=res_tot_pop2;
table res_pop;
run;

* flag any activity across all sectors;
data res_tot_pop3; set res_tot_pop2;
anyserv = anyben_Jul06Jun16 + anyhospBTadj_overnight + anyclaim_Jul06Jun16 +
anyconv_nt_Jul06Jun16 + anyscript_Jul06Jun16;
if anyserv>1 then anyserv=1;
sex1 = input(sex,8.);
drop sex;
proc freq data=res_tot_pop3; table anyserv;
run;

* wgt2 - accounts for those not in resident population and with no activity
in any of the sectors;
* official weight variable;
data res_tot_pop4(rename=(sex1=sex));
set res_tot_pop3;
wgt2 = wgt;
if anyserv=0 & res_pop=0 & wgt=1 then wgt2=0;
run;


*********************************************************************;
* create cohort specific datasets for top 10% calculation;
data coh5054; set res_tot_pop4; where cohort = 5054; run;
data coh5559; set res_tot_pop4; where cohort = 5559; run;
data coh6064; set res_tot_pop4; where cohort = 6064; run;
data coh6569; set res_tot_pop4; where cohort = 6569; run;
data coh7074; set res_tot_pop4; where cohort = 7074; run;
data coh7579; set res_tot_pop4; where cohort = 7579; run;
data coh8084; set res_tot_pop4; where cohort = 8084; run;


*****************************************
IMPORTANT: THE CODE BELOW WAS RUN FOR EACH COHORT AND THE TOTAL POPULATION.
WHEN RUNNING THE CODE, CHANGE DATASET NAME TO DO FOR EACH DATATSET;

** Macro to determine cut-points for identifying 'high-cost' individuals
(highest 10% of users in a sector)
* Calculates both highest 10% of users & 90% of services;

** Find Cut-Points based on % of SM's and % of Total;
* SEX-SPECIFIC cut-points;
%macro cut_points (area1 = , sex = , sm_pct = 10, tot_pct = 90);

      * CHANGE DATASET NAME HERE;
      proc freq data = res_tot_pop4;
            table &area1 / out = Freq_&area1 sparse;
            weight wgt2;
            where sex = &sex;
      run;
```

```
      data Freq_&area1;
            set Freq_&area1;
            length area1 $20;

            area1  = "&area1";
            value  = &area1;
            pct_sm = percent;

            if value = . then delete;
            drop &area1 percent;
      run;

      proc sort data = Freq_&area1; by area1 descending value; run;

      data  Totals
                  TotOut (keep = area1 cum_tot cum_sm);

            set Freq_&area1;
            by area1;
            sex = &sex;

            tot_outcome = count*value;

            retain cum_tot 0 cum_sm 0;

            cum_tot = cum_tot + tot_outcome;
            cum_sm  = cum_sm  + count;

            if last.area1 then output TotOut;
            output Totals;
      run;

      data  Totals1 (drop = cut_sm_&area1._&sex cut_tot_&area1._&sex)
                  Cuts_&area1._&sex (keep = cut_sm_&area1._&sex
cut_tot_&area1._&sex);
            merge Totals
                        Totout (rename = (cum_tot = grand_tot cum_sm =
grand_sm));
            by area1;

            pct_sm      = (count/grand_sm);
            pct_tot     = (tot_outcome/grand_tot);
            cum_pct_sm  = (cum_sm/grand_sm);
            cum_pct_tot = (cum_tot/grand_tot);

            retain cut_sm_&area1._&sex cut_tot_&area1._&sex;

            if cut_sm_&area1._&sex  = . and cum_pct_sm  >= &sm_pct  then
cut_sm_&area1._&sex  = value;
            if cut_tot_&area1._&sex = . and cum_pct_tot >= &tot_pct then
cut_tot_&area1._&sex = value;

            if last.area1 then output Cuts_&area1._&sex;
            output Totals1;
      run;
```

```
      proc append base = All_Totals data = Totals1; run;
%mend cut_points;


* NON-sex-specific Cut-Points;
%macro cut_point1 (area1 = , sm_pct = 10, tot_pct = 90);

      * CHANGE DATASET NAME HERE;
      proc freq data = res_tot_pop4;
            table &area1 / out = Freq_&area1 sparse;
            weight wgt2;
      run;

      data Freq_&area1;
            set Freq_&area1;
            length area1 $20;

            area1  = "&area1";
            value  = &area1;
            pct_sm = percent;

            if value = . then delete;
            drop &area1 percent;
      run;

      proc sort data = Freq_&area1; by area1 descending value; run;

      data  Totals
                  TotOut (keep = area1 cum_tot cum_sm);

            set Freq_&area1;
            by area1;

            tot_outcome = count*value;

            retain cum_tot 0 cum_sm 0;

            cum_tot = cum_tot + tot_outcome;
            cum_sm  = cum_sm  + count;

            if last.area1 then output TotOut;
            output Totals;
      run;

      data  Totals1 (drop = cut_sm_&area1 cut_tot_&area1)
                  Cuts_&area1 (keep = cut_sm_&area1 cut_tot_&area1);
            merge Totals
                        Totout (rename = (cum_tot = grand_tot cum_sm =
grand_sm));
            by area1;

            pct_sm      = (count/grand_sm);
            pct_tot     = (tot_outcome/grand_tot);
            cum_pct_sm  = (cum_sm/grand_sm);
            cum_pct_tot = (cum_tot/grand_tot);

            retain cut_sm_&area1 cut_tot_&area1;
```

```
            if cut_sm_&area1  = . and cum_pct_sm  >= &sm_pct  then
cut_sm_&area1  = value;
            if cut_tot_&area1 = . and cum_pct_tot >= &tot_pct then
cut_tot_&area1 = value;

            if last.area1 then output Cuts_&area1;
            output Totals1;
      run;

      proc append base = All_Totals data = Totals1; run;
%mend cut_point1;


** Find cut-points for all areas and permutations of interest;
** ADD IN ADDITIONAL SECTORS/VARS AS BECOME AVAILABLE;

* Males;
%cut_points (area1 = trunc_blength, sex = 1, sm_pct = .10, tot_pct = .90);
%cut_points (area1 = totlos_BTadj, sex = 1, sm_pct = .10, tot_pct = .90);
%cut_points (area1 = tot_num_claims, sex = 1, sm_pct = .10, tot_pct = .90);
%cut_points (area1 = tot_num_conv_nt, sex = 1, sm_pct = .10, tot_pct = .90);
%cut_points (area1 = tot_num_scripts, sex = 1, sm_pct = .10, tot_pct = .90);

* Females;
%cut_points (area1 = trunc_blength, sex = 2, sm_pct = .10, tot_pct = .90);
%cut_points (area1 = totlos_BTadj, sex = 2, sm_pct = .10, tot_pct = .90);
%cut_points (area1 = tot_num_claims, sex = 2, sm_pct = .10, tot_pct = .90);
%cut_points (area1 = tot_num_conv_nt, sex = 2, sm_pct = .10, tot_pct = .90);
%cut_points (area1 = tot_num_scripts, sex = 2, sm_pct = .10, tot_pct = .90);

* Full cohort;
%cut_point1 (area1 = trunc_blength, sm_pct = .10, tot_pct = .90);
%cut_point1 (area1 = totlos_BTadj, sm_pct = .10, tot_pct = .90);
%cut_point1 (area1 = tot_num_claims, sm_pct = .10, tot_pct = .90);
%cut_point1 (area1 = tot_num_conv_nt, sm_pct = .10, tot_pct = .90);
%cut_point1 (area1 = tot_num_scripts, sm_pct = .10, tot_pct = .90);



data ParetoFreqs;
      set All_Totals;

      if area1 = "trunc_blength" then area =  1;
      if area1 = "totlos_BTadj"     then area =  2;
      if area1 = "tot_num_claims"         then area =  3;
      if area1 = "tot_num_conv_nt"        then area =  4;
      if area1 = "tot_num_scripts"        then area =  5;


      Freq     = COUNT;
      PctSM    = cum_pct_sm;
      total    = tot_outcome;
      PctTotal = cum_pct_tot;

      keep sex area value freq PctSM total PctTotal;
run;
```

```
** Merge cut-points for all vars;
data all_cuts;
      merge Cuts_trunc_blength_1 Cuts_totlos_BTadj_1 Cuts_tot_num_claims_1
Cuts_tot_num_conv_nt_1 Cuts_tot_num_scripts_1
                  Cuts_trunc_blength_2 Cuts_totlos_BTadj_2
Cuts_tot_num_claims_2 Cuts_tot_num_conv_nt_2 Cuts_tot_num_scripts_2
                  Cuts_trunc_blength  Cuts_totlos_BTadj Cuts_tot_num_claims
Cuts_tot_num_conv_nt Cuts_tot_num_scripts;
run;

* find total number of individuals in dataset;
* CHANGE DATASET NAME HERE;
proc means data=res_tot_pop4; var snz_uid; run;

** Create data set to merge back into original data;
data all_cuts1;
      set all_cuts;
      do i = 1 to ########; /* Replace ######## with total number of
individuals in dataset, ADJUST AS APPROPRIATE */
            output;
      end;
      drop i;
run;

proc datasets nolist nodetails;
      delete      Totals Totals1 Totout all_cuts
                  Freq_trunc_blength Freq_totlos_BTadj Freq_tot_num_claims
Freq_tot_num_conv_nt Freq_tot_num_scripts
                  Cuts_trunc_blength_1 Cuts_totlos_BTadj_1
Cuts_tot_num_claims_1 Cuts_tot_num_conv_nt_1 Cuts_tot_num_scripts_1
                  Cuts_trunc_blength_2 Cuts_totlos_BTadj_2
Cuts_tot_num_claims_2 Cuts_tot_num_conv_nt2 Cuts_tot_num_scripts_2
                  Cuts_trunc_blength Cuts_totlos_BTad Cuts_tot_num_claims
Cuts_tot_num_conv_nt Cuts_tot_num_scripts;
run;
quit;


** Merge cut-points back into original data
* Define high/low cost groups;
* If individual is missing info on an area, we assume they are in the low-
cost group.
NO ONE CURRENTLY HAS MISSING DATA, BUT THIS MAY ARISE IN FUTURE;

data pareto_cuts;
      merge       res_tot_pop4            /* CHANGE DATASET NAME HERE */
                  all_cuts1;
      * change length of arrays throughout, as relevant;
      array area        [5]   trunc_blength  totlos_BTadj  tot_num_claims
tot_num_conv_nt  tot_num_scripts;
      array m_sm_cut    [5]   Cut_sm_trunc_blength_1  Cut_sm_totlos_BTadj_1
Cut_sm_tot_num_claims_1  Cut_sm_tot_num_conv_nt_1  Cut_sm_tot_num_scripts_1;
      array f_sm_cut    [5]   Cut_sm_trunc_blength_2  Cut_sm_totlos_BTadj_2
Cut_sm_tot_num_claims_2  Cut_sm_tot_num_conv_nt_2  Cut_sm_tot_num_scripts_2;
```

```
      array mf_sm_cut    [5]    Cut_sm_trunc_blength    Cut_sm_totlos_BTadj
Cut_sm_tot_num_claims  Cut_sm_tot_num_conv_nt  Cut_sm_tot_num_scripts;
      array hi_sm        [5]    hi_sm_ben  hi_sm_hosp  hi_sm_acc  hi_sm_conv
hi_sm_pharms;
      array hi_sm1       [5]    hi_sm_ben1  hi_sm_hosp1   hi_sm_acc1
hi_sm_conv1  hi_sm_pharms1;

      ** Define "High Cost" / "Low Cost" grouping based on 10% of SM;
      do i = 1 to 5;
            if area[i] ne . then do;
                  hi_sm[i]  = 0;
                  hi_sm1[i] = 0;
            end;

            * Based on M/F combined cut-points;
            if mf_sm_cut[i] = 0 and area[i] > 0 then hi_sm1[i] = 1;
                  else if mf_sm_cut[i] > 0 and area[i] >= mf_sm_cut[i] then
hi_sm1[i] = 1;

            * Based on sex-specific cut-points;
            if sex = 1 then do; * Males;
                  if m_sm_cut[i] = 0 and area[i] > 0 then hi_sm[i] = 1;
                        else if m_sm_cut[i] > 0 and area[i] >= m_sm_cut[i]
then hi_sm[i] = 1;
            end;

            if sex = 2 then do; * Females;
                  if f_sm_cut[i] = 0 and area[i] > 0 then hi_sm[i] = 1;
                        else if f_sm_cut[i] > 0 and area[i] >= f_sm_cut[i]
then hi_sm[i] = 1;
            end;
      end;

      ** If missing data for a sector, set high/low cost to "Low Cost";
      array all [10]    hi_sm_ben  hi_sm_hosp  hi_sm_acc  hi_sm_conv
hi_sm_pharms
                              hi_sm_ben1  hi_sm_hosp1   hi_sm_acc1
hi_sm_conv1  hi_sm_pharms1;

      do i = 1 to 10;
            if all[i] = . then all[i] = 0;
      end;


data hilo_res_tot_pop4; /* CHANGE DATASET NAME HERE */ set pareto_cuts;
label
hi_sm_ben      = "High cost, sex-specific, Benefits"
hi_sm_hosp     = "High cost, sex-specific, Hospitalization"
hi_sm_acc      = "High cost, sex-specific, ACC"
hi_sm_conv     = "High cost, sex-specific, Crime"
hi_sm_pharms   = "High cost, sex-specific, Pharmaceuticals"

hi_sm_ben1     = "High cost, cross-sex, Benefits"
hi_sm_hosp1    = "High cost, cross-sex, Hospitalization"
hi_sm_acc1     = "High cost, cross-sex, ACC"
hi_sm_conv1    = "High cost, cross-sex, Crime"
hi_sm_pharms1  = "High cost, cross-sex, Pharmaceuticals";
```

```
drop
i
Cut_sm_trunc_blength_1  Cut_sm_totlos_BTadj_1  Cut_sm_tot_num_claims_1
Cut_sm_tot_num_conv_nt_1  Cut_sm_tot_num_scripts_1
Cut_sm_trunc_blength_2  Cut_sm_totlos_BTadj_2  Cut_sm_tot_num_claims_2
Cut_sm_tot_num_conv_nt_2  Cut_sm_tot_num_scripts_2
Cut_tot_trunc_blength_1 Cut_tot_totlos_BTadj_1 Cut_tot_tot_num_claims_1
Cut_tot_tot_num_conv_nt_1 Cut_tot_tot_num_scripts_1
Cut_tot_trunc_blength_2 Cut_tot_totlos_BTadj_2 Cut_tot_tot_num_claims_2
Cut_tot_tot_num_conv_nt_2 Cut_tot_tot_num_scripts_2
Cut_sm_trunc_blength  Cut_sm_totlos_BTadj Cut_sm_tot_num_claims
Cut_sm_tot_num_conv_nt  Cut_sm_tot_num_scripts
Cut_tot_trunc_blength Cut_tot_totlos_BTadj Cut_tot_tot_num_claims
Cut_tot_tot_num_conv_nt Cut_tot_tot_num_scripts;
run;

proc means data=hilo_res_tot_pop4; run;

* save files;
data steph.hilo_coh5054; set hilo_coh5054; run;
data steph.hilo_coh5559; set hilo_coh5559; run;
data steph.hilo_coh6064; set hilo_coh6064; run;
data steph.hilo_coh6569; set hilo_coh6569; run;
data steph.hilo_coh7074; set hilo_coh7074; run;
data steph.hilo_coh7579; set hilo_coh7579; run;
data steph.hilo_coh8084; set hilo_coh8084; run;
data steph.hilo_res_tot_pop4; set hilo_res_tot_pop4; run;


* merge education data with total population;
proc sql;
      create table total_pop_educ as
            select a.*, b.cen_ind_std_highest_qual_code
            from steph.hilo_res_tot_pop4 a
                  left join census.census_individual b
                        on a.snz_uid = b.snz_uid;
quit;

** create education variable for analyses;
*     00 = no qual
      97, 99, missing = don't know (97 = response unidentifiable, 99 = not
specified)
      all else = some qual;
data total_pop_educ_2;
set total_pop_educ;
      if cen_ind_std_highest_qual_code = '00'
            then education = 'no qual';
      else if cen_ind_std_highest_qual_code = '97' or
cen_ind_std_highest_qual_code = '99' or cen_ind_std_highest_qual_code = .
            then education = 'unknown';
      else education = 'qual';
proc freq data=total_pop_educ_2; table education; run;
run;

* create multiple high user variable;
data total_pop_educ_3;
```

```
set total_pop_educ_2;
multi10 = hi_sm_acc1 + hi_sm_ben1 + hi_sm_conv1 + hi_sm_hosp1 +
hi_sm_pharms1;
proc freq data=total_pop_educ_3;
table multi10;
run;

* recode 4 or 5 high sector users into 3 - so 3 now represents 3 or more
sectors;
data total_pop_educ_4;
set total_pop_educ_3;
if multi10 = 4 or multi10 = 5 then multi10_2 = 3;
else multi10_2 = multi10;
proc freq data=total_pop_educ_4;
table multi10 multi10_2;
run;

* create total cost variable;
data total_pop_educ_5;
set total_pop_educ_4;
tot_cost_all = totcost_ben2016c + totcost_BTadjmult + totclaimcost_Jul06Jun16
+ pharms_cost;
run;

* save file;
data steph.totpop_final; set total_pop_educ_5; run;

************************;
```

```
/*************************************************************************
*************************************************************************
**********
AUTHOR: S. D'Souza
IDI REFRESH: idi_clean_20171020

PURPOSE: Analyses

DISCLAIMER:
The results in this [report, paper] are not official statistics.
They have been created for research purposes from the Integrated Data
Infrastructure (IDI), managed by Statistics New Zealand.
The opinions, findings, recommendations, and conclusions expressed in this
[report, paper etc] are those of the author(s), not Statistics NZ,
[Department X, or Organisation Y].
Access to the anonymised data used in this study was provided by Statistics
NZ under the security and confidentiality provisions of the Statistics Act
1975.
Only people authorised by the Statistics Act 1975 are allowed to see data
about a particular person, household, business, or organisation,
and the results in this [report, paper] have been confidentialised to protect
these groups from identification and to keep their data safe.
Careful consideration has been given to the privacy, security, and
confidentiality issues associated with using administrative and survey data
in the IDI.
Further detail can be found in the Privacy impact assessment for the
Integrated Data Infrastructure available from www.stats.govt.nz.

*************************************************************************
*************************************************************************
*********/

*libraries;
libname steph "/nas/DataLab/MAA/MAA2018-15 A small segment of the population
with large economic burden/Steph/Data";

** load data;
data coh5054; set steph.hilo_coh5054; run;
data coh5559; set steph.hilo_coh5559; run;
data coh6064; set steph.hilo_coh6064; run;
data coh6569; set steph.hilo_coh6569; run;
data coh7074; set steph.hilo_coh7074; run;
data coh7579; set steph.hilo_coh7579; run;
data coh8084; set steph.hilo_coh8084; run;
data total_pop; set steph.totpop_final; run;


**1. get total population & cohort counts weighted and unweighted;
* not weighted;
* the cumulative freq gives unweighted total pop count;
proc freq data=total_pop;
table cohort;
where wgt2 > 0; * not weighted but excludes all those not in resident pop;
run;

* weighted;
* cumulative freq gives weighted total pop count;
```

```
proc freq data=total_pop;
table cohort;
weight wgt2;
run;


**2. percentage with education data and percentage who left school (i.e. no
qualifications);
* total population;
proc freq data=total_pop;
table education;
weight wgt2;
run;

* by cohort;
proc sort data=total_pop;
by cohort;
proc freq data=total_pop;
table education;
by cohort;
weight wgt2;
run;

proc contents data=total_pop; run;


**3. Prevalence for each sector;
* total population;
proc freq data=total_pop;
table anyben_Jul06Jun16 anyhospBTadj_overnight anyscript_Jul06Jun16
anyclaim_Jul06Jun16 anyconv_nt_Jul06Jun16;
weight wgt2;
run;

* by cohort;
proc sort data=total_pop;
by cohort;
* males;
proc freq data=total_pop;
table anyben_Jul06Jun16 anyhospBTadj_overnight anyscript_Jul06Jun16
anyclaim_Jul06Jun16 anyconv_nt_Jul06Jun16;
by cohort;
where sex = 1;
weight wgt2;
run;

* females;
proc freq data=total_pop;
table anyben_Jul06Jun16 anyhospBTadj_overnight anyscript_Jul06Jun16
anyclaim_Jul06Jun16 anyconv_nt_Jul06Jun16;
by cohort;
where sex = 2;
weight wgt2;
run;


**4. Percentage for each sector classified as higher user - i.e. top 10%;
```

```
* total population;
proc freq data=total_pop;
table hi_sm_acc1  hi_sm_ben1  hi_sm_conv1  hi_sm_hosp1  hi_sm_pharms1;
weight wgt2;
run;


**5. Overall count and costs of events per sector for total population and
top 10%;
* totals for total population;
proc summary data=total_pop print sum ;
var trunc_blength
totlos_BTadj
tot_num_claims
tot_num_conv_nt
tot_num_scripts
totcost_ben2016c
totcost_BTadjmult
totclaimcost_Jul06Jun16
pharms_cost;
weight wgt2;
run;

** totals for top 10%;
* benefit days & costs;
proc summary data=total_pop print sum;
var trunc_blength totcost_ben2016c;
where hi_sm_ben1=1;
weight wgt2;
run;

* hospitalisations - overnight stays & costs;
proc summary data=total_pop print sum;
var totlos_BTadj totcost_BTadjmult;
where hi_sm_hosp1=1;
weight wgt2;
run;

* acc - claims & cost;
proc summary data=total_pop print sum;
var tot_num_claims totclaimcost_Jul06Jun16;
where hi_sm_acc1=1;
weight wgt2;
run;

* non-traffic convictions;
proc summary data=total_pop print sum;
var tot_num_conv_nt;
where hi_sm_conv1=1;
weight wgt2;
run;

* pharms - scripts & cost;
proc summary data=total_pop print sum;
var tot_num_scripts pharms_cost;
where hi_sm_pharms1=1;
weight wgt2;
```

```
run;


**6. Overlap analyses - odds ratios and tetrachoric correlations;
* total population;
proc freq data=total_pop;
table hi_sm_ben1*(hi_sm_hosp1 hi_sm_pharms1 hi_sm_acc1 hi_sm_conv1) / plcorr;
table hi_sm_hosp1*(hi_sm_pharms1 hi_sm_acc1 hi_sm_conv1) / plcorr;
table hi_sm_pharms1*(hi_sm_acc1 hi_sm_conv1) / plcorr;
table hi_sm_acc1*hi_sm_conv1 / plcorr;
weight wgt2;
run;

* age band by sex;
%macro overlap(cohort);
proc sort data=&cohort.;
by sex;
proc freq data=&cohort.;
by sex;
table hi_sm_ben*(hi_sm_hosp hi_sm_pharms hi_sm_acc hi_sm_conv) / plcorr;
table hi_sm_hosp*(hi_sm_pharms hi_sm_acc hi_sm_conv) / plcorr;
table hi_sm_pharms*(hi_sm_acc hi_sm_conv) / plcorr;
table hi_sm_acc*hi_sm_conv / plcorr;
weight wgt2;
run;
%mend overlap;

%overlap(coh5054);
%overlap(coh5559);
%overlap(coh6064);
%overlap(coh6569);
%overlap(coh7074);
%overlap(coh7579);
%overlap(coh8084);


**7. Negative binomial analyses - impact of education on high need groups;
* total population;
* run model for total pop, adjusting for sex and cohort;
proc genmod data=total_pop;
      class cohort sex education(ref='qual') / param=ref;
      model multi10_2 = education cohort sex / type3 dist=negbin;
            estimate 'no qual' education 1 0 / exp;
            estimate 'unknown' education 0 1 / exp;
      weight wgt2;
run;


** cohort specific, adjusted by sex;
proc sort data=total_pop;by cohort; run;
proc genmod data=total_pop;
      class sex education(ref='qual') / param=ref;
      model multi10_2 = education sex / type3 dist=negbin;
            estimate 'no qual' education 1 0 / exp;
            estimate 'unknown' education 0 1 / exp;
                  by cohort;
      weight wgt2;
```

```
run;


**8. Weighted multi sector high user counts;
* total population;
proc freq data=total_pop;
table multi10;
weight wgt2;
run;


**9. Mean years of follow up for those in resident population;
* total population;
proc means data=total_pop;
var wgt2;
where wgt2 > 0;
run;

* by cohort;
proc sort data=total_pop;
by cohort;
proc means data=total_pop;
var wgt2;
by cohort;
where wgt2 > 0;
run;


**10. correlation between benefit costs (2016c includes child dependents,
2016nc excludes child dependents);
proc corr data=total_pop;
var totcost_ben2016c totcost_ben2016nc;
weight wgt2;
run;


**11. Proportion of total costs across sectors accounted for by indivduals
who were high-need users in 3+ sectors;
* totals by multi sector high users;
proc sort data=total_pop; by multi10_2; run;

proc summary data=total_pop print sum;
var trunc_blength totcost_ben2016c totlos_BTadj totcost_BTadjmult
tot_num_claims totclaimcost_Jul06Jun16 tot_num_scripts pharms_cost
tot_num_conv_nt tot_cost_all;
by multi10_2;
weight wgt2;
run;



**12. Gini coefficients - total population and age band by sex;
*split files by sex;
%macro split (cohort);
data &cohort._m; set &cohort.; if sex=1; run;
data &cohort._f; set &cohort.; if sex=2; run;
%mend split;
```

```
%split (coh5054);
%split (coh5559);
%split (coh6064);
%split (coh6569);
%split (coh7074);
%split (coh7579);
%split (coh8084);


%macro gini (cohort, var);

proc freq data =  &cohort.;
tables &var. / noprint out = &var._&cohort.;
format &var. 7.0;
weight wgt2;
run;

data &var._&cohort._2;
set &var._&cohort.;
retain sumoff perpop;
sumoff + (&var. * count); /* cumulative hospitalisations at each point in
distribution */
perpop + percent;        /* cumulative population at each point in
distribution */
run;

proc sort data =&var._&cohort._2;
by descending sumoff ;
run;

data &var._&cohort._3;
set &var._&cohort._2;
by descending sumoff;
if _n_=1 then do;
totaloff = sumoff;
end;
retain totaloff;
peroff = (sumoff / totaloff) * 100;
run;

/* re-sort from low to high */
proc sort data = &var._&cohort._3;
by perpop;
run;

/* calculate Gini coefficient */
data &var._&cohort._4;
set &var._&cohort._3;
xlag = lag(perpop);
xlag = xlag / 100;
ylag = lag(peroff);
ylag = ylag / 100;
columna = (peroff / 100) * xlag;
columnb = (perpop / 100) * ylag;
retain suma sumb;
suma + columna;
```

```
sumb + columnb;
gini = suma - sumb;
run;

proc print data = &var._&cohort._4;
var gini;
format gini 12.10;      /* format statement added for consistency with macro
*/
where peroff = 100;
run;

%mend gini;

** NOTE: To avoid too many lines of code - I have only given the code for the
coh5054 cohorts (male and female) and total population.
      Change cohort name for other cohorts;

* males;
%gini (cohort = coh5054_m, var = trunc_blength);
%gini (cohort = coh5054_m, var = totlos_BTadj);
%gini (cohort = coh5054_m, var = tot_num_claims);
%gini (cohort = coh5054_m, var = tot_num_conv_nt);
%gini (cohort = coh5054_m, var = tot_num_scripts);

* females;
%gini (cohort = coh5054_f, var = trunc_blength);
%gini (cohort = coh5054_f, var = totlos_BTadj);
%gini (cohort = coh5054_f, var = tot_num_claims);
%gini (cohort = coh5054_f, var = tot_num_conv_nt);
%gini (cohort = coh5054_f, var = tot_num_scripts);

** total cohort;
%gini (cohort = total_pop, var = trunc_blength);
%gini (cohort = total_pop, var = totlos_BTadj);
%gini (cohort = total_pop, var = tot_num_claims);
%gini (cohort = total_pop, var = tot_num_conv_nt);
%gini (cohort = total_pop, var = tot_num_scripts);
```

```
/*************************************************************************
*************************************************************************
**********
AUTHOR: S. D'Souza
IDI REFRESH: idi_clean_20171020

PURPOSE: Analyses for revisions

DISCLAIMER:
The results in this [report, paper] are not official statistics.
They have been created for research purposes from the Integrated Data
Infrastructure (IDI), managed by Statistics New Zealand.
The opinions, findings, recommendations, and conclusions expressed in this
[report, paper etc] are those of the author(s), not Statistics NZ,
[Department X, or Organisation Y].
Access to the anonymised data used in this study was provided by Statistics
NZ under the security and confidentiality provisions of the Statistics Act
1975.
Only people authorised by the Statistics Act 1975 are allowed to see data
about a particular person, household, business, or organisation,
and the results in this [report, paper] have been confidentialised to protect
these groups from identification and to keep their data safe.
Careful consideration has been given to the privacy, security, and
confidentiality issues associated with using administrative and survey data
in the IDI.
Further detail can be found in the Privacy impact assessment for the
Integrated Data Infrastructure available from www.stats.govt.nz.

*************************************************************************
*************************************************************************
*********/

** libraries;
libname steph "/nas/DataLab/MAA/MAA2018-15 A small segment of the population
with large economic burden/Steph/Data";
libname dia ODBC dsn=idi_clean_20171020_srvprd schema=dia_clean;
libname msd ODBC dsn=idi_clean_20171020_srvprd schema=msd_clean;

* load data;
data  total_pop;
set steph.totpop_final;
run;

** 1. Mortality analyses;
* Create mortality flag and calculate days until deceased for those
who passed away during obs period;
data total_pop_2; set total_pop;
if '01JUL2006'd <= dod <= '30JUN2016'd then mort_flag = 1;
else mort_flag = 0;
if mort_flag = 1 then days_deceased = dod - '01JUL2006'd;
run;

*mortality - crosstab with multi-sector high user;
proc freq data=total_pop_2;
table multi10_2*mort_flag;
run;
```

```
*average number of days until deceased across high user groups;
proc sort data=total_pop_2;
by multi10_2;
proc means data=total_pop_2;
var days_deceased;
by multi10_2;
run;

*****************************************************************;

** 2. Redo benefit overlap analyses excluding sickness benefits;
* load benefits data;
proc sql;
      create table benefit_spells as
      select snz_uid, msd_spel_spell_start_date as start_ben format DATE10.,
                  msd_spel_spell_end_date as end_ben format DATE10.,
                  input(msd_spel_servf_code,8.) as benefit_type_num
/*converted to numeric*/
            from msd.msd_spell
                  where (msd_spel_spell_end_date = . and
msd_spel_spell_start_date <= '30JUN2016'd)
                        or ('01JUL2006'd <= msd_spel_spell_start_date <=
'30JUN2016'd)
                        or (msd_spel_spell_end_date > '30JUN2016'd and
msd_spel_spell_start_date <= '30JUN2016'd);
quit;

** Drop benefit types: student unemployment (607), invalid's (320) and
sickness (600)
* also compute first & last dates for observation period;
data benefits; set benefit_spells;
first_b = MDY (07, 01, 2006);
last_b = MDY (06, 30, 2016);
format first_b last_b date10.;
where benefit_type_num not in (607,320,600);
run;

** Flag records as right-censored;
data benefits2; set benefits;
if (end_ben ne .) and (end_ben <= '30JUN2016'd) then right_censor = 0;
else if (end_ben = .) or (end_ben > '30JUN2016'd) then right_censor = 1;

** Flag records as left-censored;
if start_ben >= '01JUL2006'd then left_censor = 0;
else if start_ben < '01JUL2006'd then left_censor = 1;

* Assign right-censored cases last date in benefits file
* Assign left-censored cases first date in benefits file;
if right_censor = 1 then end_ben = last_b;
if left_censor = 1 then start_ben = first_b;

** Determine length of benefits between 01/07/2006 - 30/06/2016;
* Scaled to days;
benefit_length = (end_ben - start_ben);
label benefit_length  = "Length of benefit (days)";
run;
```

```
** Create benefit lengths & # of spells;
proc sort data=benefits2; by snz_uid;
data benefit_flat_ALL;
      do i = 1 to 250 until (last.snz_uid);
            set benefits2;
            by snz_uid;

            retain      blength_ns_Jul06Jun16 0;

      if first.snz_uid then blength_ns_Jul06Jun16  = 0;
      end;

             blength_ns_Jul06Jun16 = blength_ns_Jul06Jun16 + benefit_length;
            anyben_ns_Jul06Jun16 = 1; /* All individuals in dataset should be
coded as '1' for any benefits */

      label anyben_ns_Jul06Jun16     = "Has 1+ benefit spell excl. sickness
and student, 01/07/2006-30/06/2016 0/1"
                  blength_ns_Jul06Jun16   = "Benefit days: 01/07/2006-
30/06/2016, excl. sickness and student";

      keep snz_uid anyben_ns_Jul06Jun16 blength_ns_Jul06Jun16;
run;

proc means data=benefit_flat_ALL; var blength_ns_Jul06Jun16; run;
* don't need to truncate this;

* merge with total pop;
proc sql;
      create table total_pop_3 as
            select * from total_pop_2 a
                  left join benefit_flat_ALL b
                  on a.snz_uid = b.snz_uid;
quit;

* give 0s to those with no benefit data;
data total_pop_4; set total_pop_3;
if anyben_ns_Jul06Jun16 ne 1 then anyben_ns_Jul06Jun16 = 0;
if anyben_ns_Jul06Jun16 = 0 then blength_ns_Jul06Jun16 = 0;
run;

** create top 10%;
* NON-sex-specific Cut-Points;
%macro cut_point1 (area1 = , sm_pct = 10, tot_pct = 90);

      * Change dataset name here;
      proc freq data = total_pop_4;
            table &area1 / out = Freq_&area1 sparse;
            weight wgt2;
      run;

      data Freq_&area1;
            set Freq_&area1;
            length area1 $20;

            area1  = "&area1";
            value  = &area1;
```

```
            pct_sm = percent;

            if value = . then delete;
            drop &area1 percent;
      run;

      proc sort data = Freq_&area1; by area1 descending value; run;

      data  Totals
                  TotOut (keep = area1 cum_tot cum_sm);

            set Freq_&area1;
            by area1;

            tot_outcome = count*value;

            retain cum_tot 0 cum_sm 0;

            cum_tot = cum_tot + tot_outcome;
            cum_sm  = cum_sm  + count;

            if last.area1 then output TotOut;
            output Totals;
      run;

      data  Totals1 (drop = cut_sm_&area1 cut_tot_&area1)
                  Cuts_&area1 (keep = cut_sm_&area1 cut_tot_&area1);
            merge Totals
                        Totout (rename = (cum_tot = grand_tot cum_sm =
grand_sm));
            by area1;

            pct_sm      = (count/grand_sm);
            pct_tot     = (tot_outcome/grand_tot);
            cum_pct_sm  = (cum_sm/grand_sm);
            cum_pct_tot = (cum_tot/grand_tot);

            retain cut_sm_&area1 cut_tot_&area1;

            if cut_sm_&area1  = . and cum_pct_sm  >= &sm_pct  then
cut_sm_&area1  = value;
            if cut_tot_&area1 = . and cum_pct_tot >= &tot_pct then
cut_tot_&area1 = value;

            if last.area1 then output Cuts_&area1;
            output Totals1;
      run;

      proc append base = All_Totals data = Totals1; run;
%mend cut_point1;


* Full cohort;
%cut_point1 (area1 = blength_ns_Jul06Jun16, sm_pct = .10, tot_pct = .90);


data ParetoFreqs;
```

```
      set All_Totals;
      Freq     = COUNT;
      PctSM    = cum_pct_sm;
      total    = tot_outcome;
      PctTotal = cum_pct_tot;

      keep value freq PctSM total PctTotal;
run;


** Create data set to merge back into original data;
data all_cuts1;
      set cuts_blength_ns_Jul06Jun16;
      do i = 1 to ########; /* Replace ######## with total number of
individuals in dataset, ADJUST AS APPROPRIATE */
            output;
      end;
      drop i;
run;

proc datasets nolist nodetails;
      delete      Totals Totals1 Totout cuts_blength_ns_Jul06Jun16
freq_blength_ns_Jul06Jun16;
run;
quit;

** Merge cut-points back into original data
* Define high/low cost groups;
* If individual is missing info on an area, we assume they are in the low-
cost group;
data pareto_cuts;
      merge       total_pop_4             /* change original dataset name
here */
                  all_cuts1;

      ** Define "High Cost" / "Low Cost" grouping based on 10% of SM;
            if blength_ns_Jul06Jun16 ne . then ben10_ns = 0;

      * Based on total pop (M/F combined) cut-points;
            if cut_sm_blength_ns_Jul06Jun16 = 0 and blength_ns_Jul06Jun16 > 0
then ben10_ns = 1;
                  else if cut_sm_blength_ns_Jul06Jun16 > 0 and
blength_ns_Jul06Jun16 >= cut_sm_blength_ns_Jul06Jun16 then ben10_ns = 1;

      ** If missing data for a sector, set high/low cost to "Low Cost";
            if ben10_ns = . then ben10_ns = 0;

data hilo_all; set pareto_cuts;
label ben10_ns = "High cost, cross-sex, non-sickness benefits";

drop cut_sm_blength_ns_Jul06Jun16 cut_tot_blength_ns_Jul06Jun16;
run;

** redo benefit analyses;

**Prevalence for each sector;
proc freq data=pareto_cuts;
```

```
table anyben_ns_Jul06Jun16;
weight wgt2;
run;

** Percentage  classified as higher user - i.e. top 10%;
proc freq data=pareto_cuts;
table ben10_ns;
weight wgt2;
run;

** Overlap analyses;
* weighted;
proc freq data=pareto_cuts;
table ben10_ns*(hi_sm_hosp1 hi_sm_pharms1 hi_sm_acc1 hi_sm_conv1) / plcorr;
weight wgt2;
run;
************************************************************;

** 3. 90/10 distribution analyses;
** getting percentiles;
*Benefits including sickness;
%macro dist(sect, var);
proc sort data=pareto_cuts; by &sect.;
proc univariate data=pareto_cuts noprint;
var &var.;
by &sect.;
weight wgt2;
output out=&sect._dist pctlpre=&var. pctlpts= 1 to 99;
run;
proc print data=&sect._dist; run;
%mend dist;

*standard benefits;
%dist (sect = hi_sm_ben1, var = totlos_BTadj);
%dist (sect = hi_sm_ben1, var = tot_num_claims);
%dist (sect = hi_sm_ben1, var = tot_num_conv_nt);
%dist (sect = hi_sm_ben1, var = tot_num_scripts);

*non-sickness benefits;
%dist (sect = ben10_ns, var = totlos_BTadj);
%dist (sect = ben10_ns, var = tot_num_claims);
%dist (sect = ben10_ns, var = tot_num_conv_nt);
%dist (sect = ben10_ns, var = tot_num_scripts);

*hospitalisations;
%dist (sect = hi_sm_hosp1, var = trunc_blength);
%dist (sect = hi_sm_hosp1, var = blength_ns_Jul06Jun16);
%dist (sect = hi_sm_hosp1, var = tot_num_claims);
%dist (sect = hi_sm_hosp1, var = tot_num_conv_nt);
%dist (sect = hi_sm_hosp1, var = tot_num_scripts);

*ACC;
%dist (sect = hi_sm_acc1, var = trunc_blength);
%dist (sect = hi_sm_acc1, var = blength_ns_Jul06Jun16);
%dist (sect = hi_sm_acc1, var = totlos_BTadj);
%dist (sect = hi_sm_acc1, var = tot_num_conv_nt);
%dist (sect = hi_sm_acc1, var = tot_num_scripts);
```

```
*crime;
%dist (sect = hi_sm_conv1, var = trunc_blength);
%dist (sect = hi_sm_conv1, var = blength_ns_Jul06Jun16);
%dist (sect = hi_sm_conv1, var = totlos_BTadj);
%dist (sect = hi_sm_conv1, var = tot_num_conv_nt);
%dist (sect = hi_sm_conv1, var = tot_num_scripts);

*pharms;
%dist (sect = hi_sm_pharms1, var = trunc_blength);
%dist (sect = hi_sm_pharms1, var = blength_ns_Jul06Jun16);
%dist (sect = hi_sm_pharms1, var = totlos_BTadj);
%dist (sect = hi_sm_pharms1, var = tot_num_claims);
%dist (sect = hi_sm_pharms1, var = tot_num_conv_nt);


** multi-high user variable by sector;
proc freq data = pareto_cuts;
table multi10_2*(hi_sm_ben1 ben10_ns hi_sm_hosp1 hi_sm_acc1 hi_sm_conv1
hi_sm_pharms1)/nocol norow nopercent;
weight wgt2;
run;

* Overall count of events for top 10% of each sector;
%macro total(sec);
proc sort data=pareto_cuts; by &sec.;
proc summary data=pareto_cuts print sum ;
var trunc_blength totlos_BTadj tot_num_claims tot_num_conv_nt
tot_num_scripts;
weight wgt2;
by &sec.;
run;
%mend total;

%total(hi_sm_ben1);
%total(hi_sm_hosp1);
%total(hi_sm_acc1);
%total(hi_sm_conv1);
%total(hi_sm_pharms1);

*save;
data steph.tot_pop_revisions_4Oct19;
set pareto_cuts; run;
```