Use of secondary prevention medications in metropolitan and non-metropolitan areas: an analysis of 41,925 myocardial infarctions in Australia

Protocol
June 28, 2023
https://github.com/cardiopharmnerd/medsremote

Adam Livori
PhD Student
adam.livori1@monash.edu

Center for Medicine Use and Safety, Faculty of Pharmacy and Pharmaceutical Sciences, Monash University, Melbourne, Australia

Contents

List of Figures

1 Preface

This is the protocol for the paper Use of secondary prevention medications in metropolitan and non-metropolitan areas: an analysis of 41,925 myocardial infarctions in Australia.
This protocol details the data preparation (cleaning) that was undertaken from a linked dataset provided by the Victorian Department of Health as the source of Victorian Admitted Episodes Dataset data for this study, and the Centre for Victorian Data Linkage (Victorian Department of Health) for the provision of data linkage. This study was approved by the Human Research and Ethics Committees from the Australian Institute for Health and Welfare (AIHW) (EO2018/4/468) and Monash University (14339).
To generate this document, the Stata package texdoc was used, which is available from: http://repec.sowi.unibe.ch/stata/texdoc/ (accessed 14 November 2022). The final Stata do file and this pdf are available here . The do file was originally coded and then exported from the Secure Unified Research Environment (SURE), and so the histograms generated were exported separately and then imported via LaTex coding due to constraints on exporting raw data from SURE. Therefore, when reproducing the code, use the do file rather than copying from the LaTex document.

2 Abbreviations

- A: Accessible (ARIA cateogry with values from 1.85 to 3.50)
- ABS: Australian Bureau of Statistics
- ACEi: Angiotensin converting enzyme inhibitor
- AF: Atrial fibrillation
- AFL: Atrial flutter
- AIHW: Australian Institute of Health and Welfare
- AMU: Acute medical unit
- ARB: Angiotensin II receptor blocker
- ARIA: Accessability/remoteness index of Australia
- ATC: Anatomical Therapeutic Chemical
- CA: Cancer
- CABG: Coronary Artery Bypass Graft
- CCU: Coronary care unit
- CPD: Chronic pulmonary disease
- DM: Diabetes melitus
- HA: Highly accessible (ARIA cateogry with values from 0.00 to 1.84)

- HF: Heart failure
- HS: Haemorrhagic stroke
- HT: Hypertension
- ICD: International classification of disease (ICD-10 Australian Modified codes were present in this dataset)
- IRSD: Index of relative socio-economic disadvantage
- IS: Ischaemic stroke
- MA: Moderately acceessible (ARIA cateogry with values from 3.51 to 5.80)
- MBS: Medicare benefits scheme
- MI: Myocardial infarction
- NDI: National Death Index
- NSTEMI: Non-ST elevation myocardial infarction
- P2Y12i: Adenosine diphosphate receptor (P2Y12) inhibitor
- PBS: Pharmaceutical Benefits Scheme
- PCI: Perceutaneous Coronary Intervention
- PDC: Proportion of days covered
- PVD: Peripheral vascular disease
- SLA: Statistical local area
- STEMI: ST elevation myocardial infarction
- US: Unknown stroke
- VAED: Victorian admitted episode dataset
- WHO: World Health Organisation

3 Introduction

Advances in access to catheter laboratories, techniques, and stent technology, have all contributed to increased survival following MI [1]. However, while this has translated into lower short-term mortality, the improved outcomes around the time of MI has led to an increase in the requirement for secondary prevention. Medications remain the mainstay of treatment for secondary prevention of subsequent MI. Randomised Control Trials (RCT) have demonstrated that there are a number of medications that reduce major adverse cardiovascular events (MACE) in MI: dual anti-antiplatelet therapy, which includes aspirin and a P2Y12 inhibitor (clopidogrel, prasugrel, or ticagrelor); angiotensin converting enzyme inhibitors/angiotensin receptor blockers (ACEI/ARB), HMG Co-A reductase inhibitors (statins); and beta blockers [1].

The utilisation of secondary prevention medications should not differ based on the location of an indivudal, yet we have seen in previous studies that remoteness can impact clinical outcomes following MI. This study posed the question as to whether secondary prevention medication dispensed and adherence may have a role to play in addressing this potential disparity in clinical outcomes.

The following protocol lists the steps were taken to create and analyse a cohort of MI admissions between 1/7/2012 and 30/6/2017 using a linked dataset from the VAED, PBS, MBS and NDI.

4 MI cohort dataset creation

4.1 Initial data import

We will generate a tag (MI Primary) that identifies those admissions with MI, check that this code correctly identifies these admissions, and then remove all other admissions. This yielded 101,850 admissions where an MI occurred at somepoint throughout the dataset time period.

```
set rmsg on
cd "G:\Adam\Project 2  - location and MI outcomes\Data"
use "G:/Jed/VAED.dta", clear
br
keep ppn agegroup sex sameday los_type hithsep admdate mm_admdate yy_admdate sepdate mm_sepdate yy_s
> epdate los sepmode admsourc drg2 tdiag1-tdiag40 toper1-toper40 sla region_r lga state
gen MI_primary = 1 if inrange(substr(tdiag1,1,3), "I21", "I2299")
br MI_primary tdiag1 if MI_primary !=1
bysort ppn : egen MI_present = min(MI_primary)
br ppn tdiag1 MI_primary MI_present
keep if MI_present == 1
ta MI_present MI_primary
rename admdate admdate_duplicate
gen admdate = date(admdate_duplicate, "DMY####")
format admdate %td
br admdate admdate_duplicate
rename sepdate sepdate_duplicate
gen sepdate = date(sepdate_duplicate, "DMY####")
format sepdate %td
br sepdate sepdate_duplicate
count if month(admdate) != mm_admdate
count if year(admdate) != yy_admdate
count if month(sepdate) != mm_sepdate
count if year(sepdate) != yy_sepdate
drop mm_admdate mm_sepdate yy_admdate yy_sepdate admdate_duplicate sepdate_duplicate
hist admdate if MI_primary == 1, color(black) graphregion(color(white)) frequency title(Histrogram o
> f MI admissions) xtitle(MI admission date)
graph export "G:\Adam\Project 2  - location and MI outcomes\Results\HIST MIadmissionsALL.pdf", as(pd
> f) name("Graph")
save MI_cohort_raw, replace
```

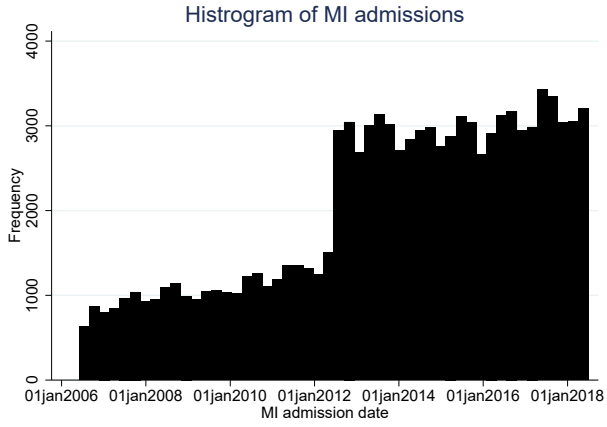

Figure 4.1: Histogram of all MI admissions from VAED dataset

4.2 Removal of nested admissions

```
use MI_cohort_raw, clear
ta sepmode if MI_primary == 1
```

As expected within this cohort, the majority of discharges are to home (68%) or transfered to another hospital (26%). In order to remove the nested admissions, such as patients going to dialysis or from CCU to AMU, we need to combine (comb) different separations to create broader categories which we can then drop follwoing extraction of relevant data. This ensures we keep the correct number of MIs (removal of nested admissions) but do not lose any data from within those admissions (comorbidities and procedure codes for example)

- A Separation and transfer to mental health residential facility
- B Separation and transfer to Transition Care bed based program
- D Death
- E Change to Interim Care
- F Change to Interim Care - Nursing Home type
- G Posthumous Organ Procurement
- H Separation to private residence/accommodation
- K Other formal separation
- M Dis to home with ref to CRC arranged
- N Separation and transfer to aged care residential facility
- O Other change in care type not BPX or F
- P Dis to home w Post Acute Care arranged
- R Separation and Transfer to Restorative Care Bed-Based Program
- S Statistical Separation
- T Separation and transfer to acute hospital/extended care
- U Dis to home w District Nursing arranged
- V Transfer to oth hospital in Region
- X Change from oth care type to NHT care
- Z Left against medical advice

For our puposes, we will use the following

- Sepmode $\bar{1}$ means a statistical discharge or transfer to other acute site

- Sepmode $\bar{2}$ means not a statistical discharge
- Sepmode $\bar{3}$ means death

The following code generates a variable if the admission possessed a primary diagnosis of MI (MI primary) and if the admission occurred prior to a separation or if no separation exists for the admission. We can then change the MI primary value to 1 should an MI occur in a nested admission.

```
rename sepmode sepmode_comb
gen sepmode = 1 if sepmode_comb =="S" | sepmode_comb == "T"
replace sepmode = 3 if sepmode_comb == "D"
replace sepmode = 2 if sepmode_comb != "S" & sepmode_comb != "T" & sepmode_comb != "D"
ta sepmode sepmode_comb if MI_primary == 1
drop sepmode_comb
br ppn sepmode tdiag1 MI_primary admdate sepdate
quietly {
forval i = 1/60 {
noisily di `i´
bysort ppn (admdate sepdate sepmode) : gen nghost  = 1 if admdate < sepdate[_n-1] & sepdate[_n-1]!=.
bysort ppn (admdate sepdate sepmode) : gen MI_nest = MI_primary[_n+1] if nghost[_n+1]==1
recode MI_primary .=1 if MI_nest == 1
bysort ppn (admdate sepdate sepmode) : gen admdate_nest = admdate[_n+1] if nghost[_n+1]==1
format admdate_nest %td
```

The next step replaces the admdate in the row with the actual date of MI admission should it occur in a nested admission of an episode of care. This alters the actual admission date, but becuase we are focussing on MI admissions, it is okay for this to be replaced, as it will not change the admission date.
We then duplicated the process but for the separation (discharge) date. It is also important to ensure the sepdate is updated should the separation date of the nexted admission be later than the original sepdate

```
replace admdate = admdate_nest if MI_nest == 1 & MI_primary ==.
bysort ppn (admdate sepdate sepmode) : gen sepdate_nest = sepdate[_n+1] if nghost[_n+1]==1
format sepdate_nest %td
replace sepdate = sepdate_nest if sepdate_nest > sepdate & sepdate_nest!=. & MI_primary == 1
```

The last step is to replace the mode of separation (type of discharge). As before, we first create a new variable that will be a copy of the sepmode if a nested admission occurred. Now that a copy variable of sepmode exists, we want to replace the sepmode providing the nested separation date is after the original separation and that the admission possessed an MI (from when we recoded them previously)

```
bysort ppn (admdate sepdate sepmode) : gen sepmode_nest = sepmode[_n+1] if nghost[_n+1]==1
replace sepmode = sepmode_nest if sepdate_nest > sepdate & sepdate_nest!=. & MI_primary == 1
```

colorblack This recode step ensures that statistical separations and transfers are recoded to actual non-death separations if this was the final outcome. The last step ensures if death occurred at any point throughout the separation, that the outcome is noted to be death.

```
recode sepmode 1=2 if sepdate_nest==sepdate & sepmode_nest == 2
replace sepmode = 3 if sepmode_nest == 3
noisily drop if nghost == 1
drop nghost MI_nest admdate_nest sepdate_nest sepmode_nest
}
}
ta sepmode if MI_primary == 1
hist admdate if MI_primary == 1, color(black) graphregion(color(white)) frequency title(Histrogram o
> f MI admissions) xtitle(MI admission date)
graph export "G:\Adam\Project 2  - location and MI outcomes\Results\HIST MIadmissionsNONEST.pdf", as
> (pdf) name("Graph")
save MI_cohort_nest_free, replace
```

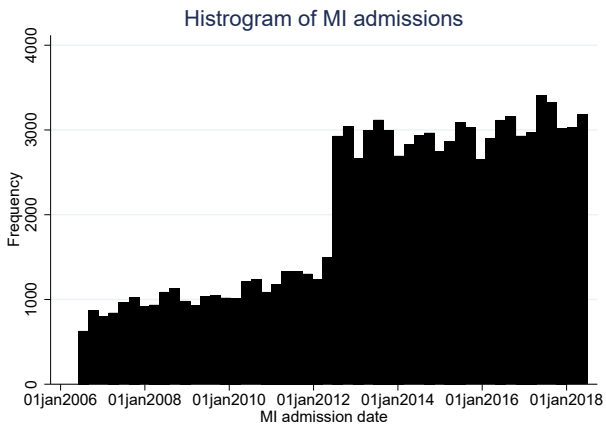

Figure 4.2: Histogram of all MI admissions from VAED dataset following removal of nested admissions

4.3 Removal of same-day admissions

```
use MI_cohort_nest_free, clear
bysort ppn admdate sepdate (sepmode) : gen nghost = _n
bysort ppn admdate sepdate (sepmode) : gen Nghost = _N
br ppn sepmode tdiag1 MI_primary admdate sepdate nghost Nghost if Nghost !=1
bysort ppn admdate sepdate (sepmode) : gen MI_sd = MI_primary[_n+1] if admdate==admdate[_n+1] & sepd
> ate==sepdate[_n+1]
recode MI_primary .=1 if MI_sd == 1
bysort ppn admdate sepdate (sepmode) : gen sepmode_sd = sepmode[_n+1] if admdate==admdate[_n+1] & se
> pdate==sepdate[_n+1]
recode sepmode 1=2 if sepmode_sd == 2
replace sepmode = 3 if sepmode_sd == 3
keep if nghost == 1
ta MI_primary
drop nghost Nghost MI_sd sepmode_sd
save MI_cohort_nest_sd_free, replace
```

4.4 Removal of transfers as separations

We specifed that a transfer is likely to occur when the separation for transfer or statistical discharge (sepmode =1) occurs within 5 days.

```
use MI_cohort_nest_sd_free, clear
gen transfer_potential = 1 if sepmode == 1 & MI_primary == 1
gen transfer = 0
bysort ppn (admdate sepdate sepmode) : replace transfer = 1 if transfer_potential==1 & (sepdate+5 >
> admdate[_n+1])
forval i = 1/60 {
replace MI_primary = 1 if transfer[_n-1]==1 & MI_primary[_n-1]==1
bysort ppn (admdate sepdate sepmode) : replace transfer = 1 if admdate[_n+1]==sepdate & MI_primary =
> = 1
}
save MI_cohort_nest_tf_free, replace
use MI_cohort_nest_tf_free, clear
gen D2S_count=0
gen D2S=.
sort ppn admdate sepdate sepmode
```

The following loop searches through all admissions marked as transfer (from the prior transfer identification code) and replaces the D2S count variable with total consecutive coutned admissions

that occur within 5 days of the previous separation. The second stage replaces the value of D2S with the maximum count of separations that occur within what has been definied as an index MI admission (five day rule between transfers).
We only keep discrete MI admissions by keeping the index admission where an MI occurred (this will still contain the total admission period due to the code generated prior to this step.

```
forval i = 0/46 {
replace D2S_count = D2S_count+1 if transfer[_n+`i´]==1
replace D2S = D2S_count if transfer[_n+`i´]==0 & D2S==. & transfer[_n-1]==0
}
forval i = 0/46 {
replace sepmode = sepmode[_n+`i´] if D2S== `i´
replace sepdate = sepdate[_n+`i´] if D2S== `i´
}
br ppn sepmode MI_primary tdiag1 admdate sepdate transfer transfer_potential D2S
keep if D2S!=.
keep if MI_primary == 1
save MI_nest_sd_tf_complete, replace
```

4.5 National death index cleaning and merge

Using the dataset that is free of nested admissions and same-day admissions, we can generate a tag to note if the person had been admitted for MI previously. Because transfers have also been accounted for, we can recode the transfers and statistical admissions (sepmode=1) as discharges.

```
use MI_nest_sd_tf_complete, clear
keep ppn agegroup sex admdate sepdate sepmode sla lga region_r state
bysort ppn (admdate) : gen priorMI = _n-1
*
recode sepmode 1=2
replace sepmode = sepmode-1
```

We now merged in the NDI data set, dropping deaths that are not part of the MI cohort or deaths that occurred before the admission date.
We also created a dataset for use in clinical outcome assessments, however these outcomes were not part of this specific study.

```
merge m:1 ppn using NDI
drop if _merge==2
count if deathdate < admdate
drop if deathdate < admdate
save MI_NDI_cause, replace
drop _merge othercauses underlying_cause_of_death
```

Of note, individuals who are under the age of 30 are not included within the dataset, irrespective if they had an MI during the dataset time period. However, if the indivudal has a subsequent cardiac event during the dataset time period and are now over the ago 30, they are included. As such, any individuals under the age of 30 needed to be removed in order to prevent incomplete data being subject to analysis.

```
gen young = 1 if agegroup == "20-24" | agegroup == "25-29"
ta young
drop if young== 1
drop young
hist admdate if admdate >= td(1/7/2012), color(black) graphregion(color(white)) frequency title(Hist
> rogram of MI admissions) xtitle(MI admission date)
graph export "G:\Adam\Project 2  - location and MI outcomes\Results\HIST MIadmissionsPOSTTF.pdf", as
> (pdf) name("Graph")
gen los = sepdate-admdate
su los

hist los if los < 100, bin(100) color(black) graphregion(color(white)) frequency title(Histrogram of
>  length of stay (if under 100 days))
```

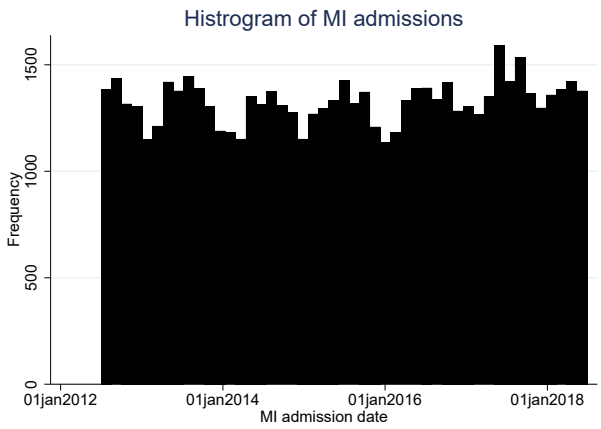

Figure 4.3: Histogram of MI admissions from VAED dataset following removal of nested admissions and transfers

```
graph export "G:\Adam\Project 2  - location and MI outcomes\Results\HIST MIadmissionsLOS.pdf", as(pd
> f) name("Graph")


*Save with location data present for use in location analysis
save MI_cohort_ndi_location, replace
*Then save cohort for cormorbidity and medicaiton analysis
drop sla lga region_r state
save MI_cohort_ndi, replace
```

4.6 Comorbidity assignment

The use of assigning comorbidities were used both to describe the study cohort, but also for creation of co-variates to be used in the regression modelling. This step involved splitting the cohort into a series of datasets so that each individual only has one attrituble MI within each dataset (leanMI). We then merged in the dataset that contains all admissions and ICD diagnosis codes. We ran the merged set (matches only) through a loop to tag the presence of the following comorbidities via the presence of ICD codes:

- AF: Atrial fibrillation
- AFL: Atrial flutter
- ARB: Angiotensin II receptor blocker
- CA: Cancer
- CPD: Chronic pulmonary disease
- DM: Diabetes melitus
- HF: Heart failure
- HS: Haemmoraghic stroke

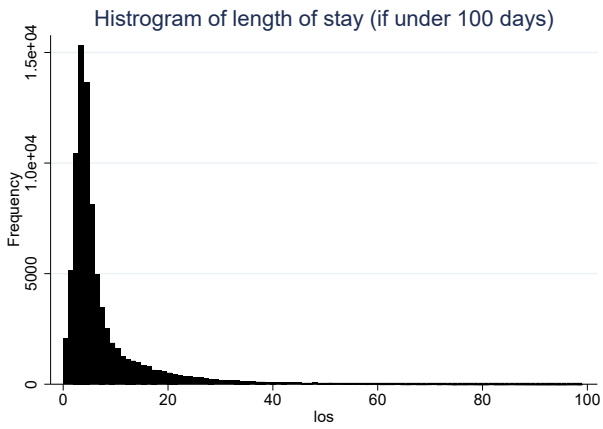

Figure 4.4: Histogram of length of stay for MI admissions (with outlying admissions greater than 100 days filtered out)

- HT: Hypertension
- IS: Ischaemic stroke
- MI: Myocardial infarction
- NSTEMI: Non-ST elevation myocardial infarction
- PVD: Peripheral vascular disease
- STEMI: ST elevation myocardial infarction
- US: Unknown stroke

Given the majority of strokes are ischaemic, we combined unknown strokes into the ischaemic stroke set as well.

```
use MI_cohort_ndi, clear
bysort ppn (admdate) : gen nghost=_n
ta nghost
forval i = 1/16 {
use MI_cohort_ndi, clear
bysort ppn (admdate) : keep if _n == `i´
keep ppn admdate sepdate
save leanMI_`i´, replace
}
forval i=1/16 {
use leanMI_`i´, clear
rename (admdate sepdate) (lean_admdate lean_sepdate)
merge 1:m ppn using MI_cohort_nest_tf_free
keep if _merge==3
drop _merge
keep if (admdate >= lean_admdate & sepdate <= lean_sepdate) | (sepdate==lean_sepdate) | (admdate==le
> an_admdate)
gen STEMI = 0
gen NSTEMI = 0
gen HT = 0
gen AF = 0
gen AFL = 0
```

```
gen HF = 0
gen IS = 0
gen HS = 0
gen US = 0
gen DM = 0
gen PVD = 0
gen CA = 0
gen CPD = 0

forval j=1/40 {
replace STEMI = 1 if substr(tdiag`j´,1,4) == "I210" |  substr(tdiag`j´,1,4) == "I211" | substr(tdiag
> `j´,1,4) == "I212" | substr(tdiag`j´,1,4) == "I213"
replace NSTEMI = 1 if inrange(tdiag`j´, "I214","I219")
replace HT = 1 if inrange(tdiag`j´, "I10","I1599") | substr(tdiag`j´,1,4)=="U823"
replace AF = 1 if inrange(tdiag`j´, "I48","I4899")
replace AFL =1 if substr(tdiag`j´,1,4) == "I482" | substr(tdiag`j´,1,4) == "I483" |substr(tdiag`j´,1
> ,4) == "I484"
replace HF = 1 if substr(tdiag`j´,1,4) == "I099" | substr(tdiag`j´,1,4) == "I110" | substr(tdiag`j´,
> 1,4) == "I130" | substr(tdiag`j´,1,4) == "I132" | substr(tdiag`j´,1,4) == "I255" | inrange(tdiag`j
> ´, "I420","I422") | inrange(tdiag`j´, "I425","I4299") | substr(tdiag`j´,1,3) == "I43" | substr(tdi
> ag`j´,1,3) == "I50" | substr(tdiag`j´,1,4) == "P290"
replace IS = 1 if inrange(tdiag`j´, "I60","I6099") | inrange(tdiag`j´, "I61","I6199") | inrange(tdia
> g`j´, "I62","I6299")
replace HS = 1 if inrange(tdiag`j´, "I63","I6399") | inrange(tdiag`j´, "G45","G4599") | inrange(tdia
> g`j´, "G46","G4699")
replace US = 1 if substr(tdiag`j´,1,3) == "I64"
replace DM =1 if inrange(tdiag`j´,"E10","E1499")
replace PVD = 1 if inrange(tdiag`j´,"I70","I7099") | inrange(tdiag`j´,"I71","I7199") | inrange(tdiag
> `j´,"I731","I7399") | inrange(tdiag`j´,"I74","I7499") | substr(tdiag`j´,1,4) == "I790" | substr(td
> iag`j´,1,4) == "I792" | substr(tdiag`j´,1,4) == "K551" | inrange(tdiag`j´, "K558","K5599") | inran
> ge(tdiag`j´, "Z958","Z9599")
replace CA = 1 if inrange(tdiag`j´,"C00","C2699") | inrange(tdiag`j´,"C30","C3499") | inrange(tdiag`
> j´,"C37","C4199") | inrange(tdiag`j´,"C43","C4499") | inrange(tdiag`j´,"C45","C5899") | inrange(td
> iag`j´,"C60","C7699") | inrange(tdiag`j´,"C81","C8599") | substr(tdiag`j´,1,3) == "C88" | inrange(
> tdiag`j´,"C90","C9799") | inrange(tdiag`j´,"C77","C80999")
replace CPD = 1 if inrange(tdiag`j´,"J40","J4499") | substr(tdiag`j´,1,4) == "J684" | substr(tdiag`j
> ´,1,4) == "J701" | substr(tdiag`j´,1,4) == "J703" | substr(tdiag`j´,1,4) == "J961"
}
collapse (sum) STEMI - CPD, by(ppn lean_admdate)
rename lean_admdate admdate
save MI_comorbid_`i´, replace
}
forval i=1/16 {
append using MI_comorbid_`i´
}
foreach i of varlist STEMI-CPD {
replace `i´ = 1 if `i´ > 1
}
rename IS IS_noUS
gen IS = 0
replace IS = 1 if IS_noUS==1 | US==1
save MI_comorbid, replace
ta STEMI
ta NSTEMI
ta HT
ta AF
ta AFL
ta HF
ta IS
ta HS
ta US
ta DM
ta PVD
ta CA
ta CPD
```

4.7 Treatment outcomes

Treatments of interest included PCI and CABG, which will both be used as an adjustment variables in the regression analysis and for description of the cohort in terms of revascularisation recevied within 30 days of the index MI.

```
use MI_cohort_raw, clear
br
keep ppn admdate sepdate toper1-toper40
gen CABG =.
gen PCI =.
quietly {
forval i = 1/40 {
replace CABG = sepdate if inrange(toper`i´,3849700,3849707) | inrange(toper`i´,3850000,3850305)
replace PCI = sepdate if inrange(toper`i´,3830600,3830605) | toper`i´==3830000 |toper`i´==3830300 |
> toper`i´==3830301 | inrange(toper`i´, 3830900,3831801) | inrange(toper`i´,9021800,9021802)
}
}
format CABG %td
format PCI %td
count if CABG !=.
count if PCI !=.
hist CABG if CABG > td(1/7/2012), bin(100) color(black) graphregion(color(white)) frequency title(CA
> BGs performed across total dataset)
graph export "G:\Adam\Project 2  - location and MI outcomes\Results\HIST MIadmissionsCABGs.pdf", as(
> pdf) name("Graph")
```

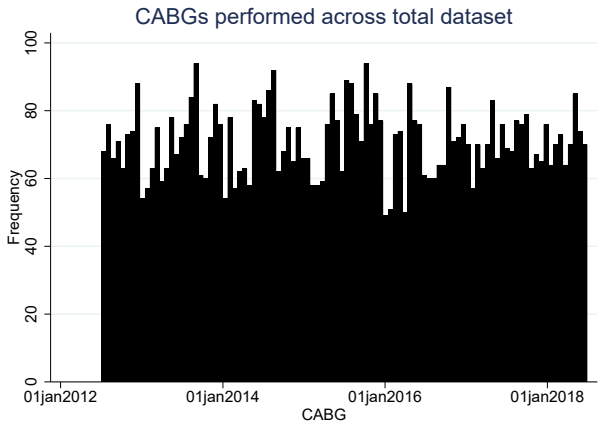

Figure 4.5: Histogram of CABG performed within 30 days of index MI admission

```
hist PCI if PCI > td(1/7/2012), bin(100) color(black) graphregion(color(white)) frequency title(PCIs
>  performed across total dataset)
graph export "G:\Adam\Project 2  - location and MI outcomes\Results\HIST MIadmissionsPCIs.pdf", as(p
> df) name("Graph")
keep ppn admdate CABG PCI
save MI_proc, replace


quietly {
forval i = 1/16 {
use MI_cohort_ndi, clear
bysort ppn (admdate) : keep if _n == `i´
merge 1:m ppn using MI_proc
keep if _merge==3
keep ppn admdate sepdate CABG PCI
drop if CABG < admdate
drop if PCI <admdate
```

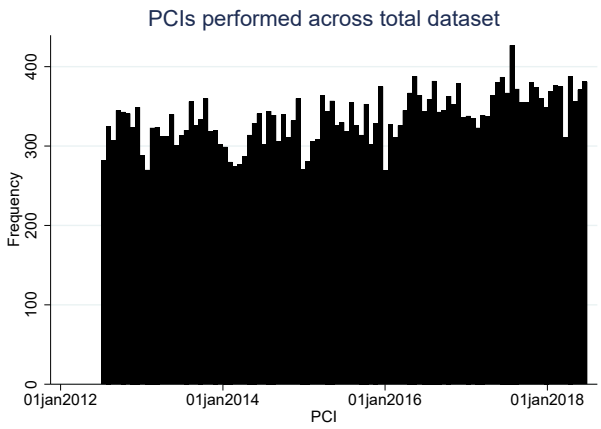

Figure 4.6: Histogram of PCI performed within 30 days of index MI admission

```
save MI_proc_merge`i´, replace
}
}
clear
forval i = 1/16 {
append using MI_proc_merge`i´
}
br
count if CABG < admdate
count if PCI < admdate
bysort ppn admdate (CABG PCI) : keep if _n==1
keep if inrange(sepdate,td(1/7/2012),td(30/6/2017))
replace CABG =. if CABG >= sepdate + 90
replace PCI =. if PCI >= sepdate + 90
count if CABG > sepdate+90 & CABG !=.
count if PCI > sepdate+90 & PCI !=.
gen CABG_tag = 0
replace CABG_tag = 1 if CABG !=. & CABG - sepdate <= 90 | CABG < admdate + 90
ta CABG_tag
gen PCI_tag = 0
replace PCI_tag = 1 if PCI !=. & PCI - sepdate <= 90 | PCI < admdate + 90
ta PCI_tag
save MI_proc_clean, replace
```

4.8 Linking statistical local area (SLA) with remoteness (ARIA)

A dataset was sourced from the AIHW (Catalogue number PHE 53) and converted from a PDF to csv file. This file lists SLA codes present in the MI cohort and allows matching with the mean ARIA values for regression analysis and attributing the broad ARIA categories for descriptive statistics.

```
use MI_cohort_ndi_location, clear
keep if inrange(sepdate,td(1/7/2012),td(30/6/2017))
drop agegroup sex lga region_r sepmode priorMI deathdate
br
ta state
save MI_sla_location_merge_prep, replace
use MI_sla_location_merge_prep, clear
import excel "G:\Adam\Resources and codes\SLA to ARIA dataset.xlsx", firstrow clear
br
rename code sla
save sla_location_key, replace
```

```
use MI_sla_location_merge_prep, clear
merge m:1 sla using sla_location_key
```

There were 4027 MIs not matched, equating to a loss of data of 9

```
keep if _merge != 2
save MI_sla_location_merge, replace
ta state
keep if state == 2
drop if Mean ==.
drop Min Max RRMAClass _merge state los SLAname sepdate
save MI_sla_match_VIC_noblanks, replace
```

4.9 Removal of admissions with in-hopsital and 90-day mortality

```
use MI_cohort_ndi, clear
keep if inrange(sepdate,td(1/7/2012),td(30/6/2017))
gen death_time = deathdate-sepdate
gen hosp_mort = 0
replace hosp_mort = 1 if death_time < 1
replace hosp_mort = 2 if death_time >= 1 & death_time <=90
histogram death_time, bin(100) color(black) graphregion(color(white)) frequency title(Days until dea
> th from separation (if died prior to 30/6/2018)) xtitle(Days to death from separation)
graph export "G:\Adam\Project 2  - location and MI outcomes\Results\HIST daysuntildeath.pdf", as(pdf
> ) name("Graph")
ta hosp_mort
```

Figure 4.7: Histogram of days until death from cohort where mortality occurred between index MI admission and end of follow up (30/6/2018)

5 Medication dataset creation

5.1 PBS data preparation- antiplatelets

This step involved taking all medications within the ATC 1st level code "B" that includes antithrombotic agents such as P2Y12 inhibitors and aspirin. We reformatted the dates so they could be used for data cleaning, and removed all observations that occurred outside the study period and observations that were not of the target drugs: aspirin, clopidogrel, ticagrelor or prasugrel. We also needed to use a PBS code for identification of a combination product containing clopidogerl

and aspirin, which was nested under clopidogrel.
This dataset was then saved so it could be merged into the MI cohort for medication use analysis.

```
use "G:\Jed\PBS B", clear
br
gen supplydate = date(date_of_supply, "DMY")
format supplydate %td
gen prescribedate = date(date_of_prescribing, "DMY")
format prescribedate %td
drop date_of_supply date_of_prescribing
drop if supplydate<td(1/4/2012)
gen aspirin =0
replace aspirin = 1 if atc=="B01AC06"
ta aspirin
gen P2Y12 = 0
replace P2Y12 = 1 if atc=="B01AC04" | atc=="B01AC22" | atc=="B01AC24" | pbs_item_code=="09296G"
ta P2Y12
ta aspirin P2Y12
gen clopi = 0
replace clopi = 1 if atc=="B01AC04" | pbs_item_code=="09296G"
gen pras = 0
replace pras = 1 if atc=="B01AC22"
gen tica = 0
replace tica = 1 if atc=="B01AC24"
ta tica P2Y12
ta pras P2Y12
ta clop P2Y12
keep if aspirin == 1 | P2Y12 == 1
save PBS_B, replace
```

5.2 PBS data preparation- non antiplatelet secondary prevention medications

The same process as before was undertaken, but this time using dispensing data from the cardiovascular ATC 1st level code "C". Drug classes included all beta blockers, ACEi, ARB, and statins. Combination products containing these ingredients were also included under their respective classes. A secondary check was then performed to ensure no drugs were classified twice.

```
use "G:\Jed\PBS C", clear
br
gen supplydate = date(date_of_supply, "DMY")
format supplydate %td
gen prescribedate = date(date_of_prescribing, "DMY")
format prescribedate %td
drop date_of_supply date_of_prescribing
drop if supplydate<td(1/4/2012)
ta atc
gen beta = 0
replace beta = 1 if inrange(substr(atc,1,4), "C07A" , "C07F")
gen ace = 0
replace ace = 1 if inrange(substr(atc,1,4), "C09A" , "C09B")
gen arb = 0
replace arb = 1 if inrange(substr(atc,1,4),"C09C","C09D")
gen statin = 0
replace statin = 1 if inrange(substr(atc,1,7), "C10AA01", "C10AA08") | inrange(substr(atc,1,5), "C10
> BA", "C10BX")
gen ace_arb = 0
replace ace_arb = 1 if ace == 1 | arb == 1
keep if beta == 1 | ace == 1 | arb == 1 | statin == 1 | ace_arb == 1
gen drugcheck = (beta+ace+arb+statin)
ta drugcheck
drop drugcheck
save PBS_C, replace
```

5.3 PBS data merge with MI cohort

We adopted a similar approach to the co-morbidity dataset creation by separating out the 'lean MIs' and merging each of the 16 datasets with the PBS data, matching for the patient identifier (ppn). This process was done for both PBS sets B and C, and then appended to create one dataset containing all PBS dispensings of secondary prevention medications that were dispensed a year prior to admission and the year following discharge for the total MI cohort.

```
use MI_cohort_ndi, clear
br
quietly {
forval i = 1/16 {
use MI_cohort_ndi, clear
bysort ppn (admdate) : keep if _n == `i´
merge 1:m ppn using PBS_B
keep if _merge==3
keep if inrange(supplydate, admdate-365, sepdate+365)
drop _merge
save MI_cohort_ndi_drug_B`i´, replace
}
}
forval i= 1/16 {
append using MI_cohort_ndi_drug_B`i´
}
bysort ppn admdate : gen nghost = _n
count if nghost == 1
br
count if nghost==1 & inrange(sepdate,td(1/7/2012),td(30/6/2017))
drop nghost
use MI_cohort_ndi, clear
br
quietly {
forval i = 1/16 {
use MI_cohort_ndi, clear
bysort ppn (admdate) : keep if _n== `i´
merge 1:m ppn using PBS_C
keep if _merge==3
keep if inrange(supplydate, admdate-365, sepdate+365)
drop _merge
save MI_cohort_ndi_drug_C`i´, replace
}
}
clear
forval i= 1/16 {
append using MI_cohort_ndi_drug_C`i´
}
br
bysort ppn admdate : gen nghost = _n
count if nghost == 1
br
count if nghost==1 & inrange(sepdate,td(1/7/2012),td(30/6/2017))
drop nghost
append using MI_cohort_ndi_PBS_B
br
ta priorMI
```

Following creation of this dataset, we removed any observations relating to MI admissions prior to 1/7/2012 and after 30/6/2017, and created a number of tags to allow identification of medication dispensing and class. Tags were created if a supply occurred within 90 days of discharge and for 30 days prior to admission.

```
use MI_cohort_ndi_PBS, clear
keep if inrange(sepdate,td(1/7/2012),td(30/6/2017))
gen supplied_90d = 0
replace supplied_90d = 1 if inrange(supplydate, sepdate, sepdate+90)
br
```

```
gen supplied_priorMI = 0
replace supplied_priorMI = 1 if inrange(supplydate, admdate-30, admdate)
ta supplied_priorMI supplied_90d
drop supplied_priorMI
gen supplied_priorMI = 0
replace supplied_priorMI = 1 if inrange(supplydate, admdate-30, admdate-1)
ta supplied_priorMI supplied_90d
br
bysort ppn sepdate (atc) : gen nghost=_n
gen beta_90d = 0
replace beta_90d = 1 if beta==1 & supplied_90d ==1
gen ace_90d = 0
replace ace_90d = 1 if ace==1 & supplied_90d ==1
gen arb_90d = 0
replace arb_90d = 1 if arb==1 & supplied_90d ==1
gen P2Y12_90d = 0
replace P2Y12_90d = 1 if P2Y12==1 & supplied_90d ==1
gen aspirin_90d = 0
replace aspirin_90d = 1 if aspirin==1 & supplied_90d ==1
gen statin_90d = 0
replace statin_90d = 1 if statin==1 & supplied_90d ==1
gen ace_arb_90d = 0
replace ace_arb_90d = 1 if (ace==1 & supplied_90d ==1 )| (arb==1 & supplied_90d ==1)
br
save MI_drugs_all, replace
```

5.4 People alive at 90 days post discharge

Because we have now have a dataset containing mortality status from the NDI and medication information from the PBS, we are able to identify those who received drugs but died during admission or within 90 days of discharge. This dataset will be needed for the PDC calculations.

```
use MI_drugs_all, clear
br
gen alive_time = deathdate-sepdate
count if alive_time < 0
gen alive_90d = 1
replace alive_90d=0 if alive_time <= 90
ta alive_90d
drop if alive_90d == 0
save MI_drugs_alive90d, replace
```

5.5 Packsize information from the PBS data

We performed a webscrape of the PBS data from the PBS website . This allowed review of potential packsize changes across the study period and to account for it should it occur. Due to slight changes in formatting of archived data, the coding changed for groups of different time periods as shown below. Once all files were extracted, they were tagged for year and month and then appended into one reference file.

```
foreach i in 2010 {
forval j = 1/12 {
if `j´ < 10 {
copy "https://www.pbs.gov.au/downloads/`i´/`i´-0`j´-01-general-schedule-ascii.zip" PBS_TF_`i´_`j´.zi
> p, replace
}
else {
copy "https://www.pbs.gov.au/downloads/`i´/`i´-`j´-01-general-schedule-ascii.zip" PBS_TF_`i´_`j´.zip
> , replace
}
}
}

copy "https://www.pbs.gov.au/downloads/2011/01/2011-01-01-general-schedule-ascii.zip" PBS_TF_2011_1.
```

```
> zip, replace
copy "https://www.pbs.gov.au/downloads/2011/01/2011-02-01-general-schedule-ascii.zip" PBS_TF_2011_2.
> zip, replace
foreach i in 2011 {
forval j = 3/12 {
if `j´ < 10 {
copy "https://www.pbs.gov.au/downloads/`i´/0`j´/`i´-0`j´-01-general-schedule-ascii.zip" PBS_TF_`i´_`
> j´.zip, replace
}
else {
copy "https://www.pbs.gov.au/downloads/`i´/`j´/`i´-`j´-01-general-schedule-ascii.zip" PBS_TF_`i´_`j´
> .zip, replace
}
}
}

foreach i in 2012 {
foreach j in 1 3 4 5 6 7 8 9 10 {
if `j´ < 10 {
copy "https://www.pbs.gov.au/downloads/`i´/0`j´/`i´-0`j´-01-general-schedule-ascii.zip" PBS_TF_`i´_`
> j´.zip, replace
}
else {
copy "https://www.pbs.gov.au/downloads/`i´/`j´/`i´-`j´-01-general-schedule-ascii.zip" PBS_TF_`i´_`j´
> .zip, replace
}
}
}
*no PBS update in November 2012
copy "https://www.pbs.gov.au/downloads/2012/12/2012-12-01-extracts.zip" PBS_TF_2012_12.zip, replace
foreach i in 2013 {
foreach j in 1 2 3 4 5 8 9 10 11 12 {
if `j´ < 10 {
copy "https://www.pbs.gov.au/downloads/`i´/0`j´/`i´-0`j´-01-extracts.zip" PBS_TF_`i´_`j´.zip, replac
> e
}
else {
copy "https://www.pbs.gov.au/downloads/`i´/`j´/`i´-`j´-01-extracts.zip" PBS_TF_`i´_`j´.zip, replace
}
}
}
{
copy "https://www.pbs.gov.au/publication/schedule/2013/06/2013-06-01-extracts.zip" PBS_TF_2013_6.zip
> , replace
copy "https://www.pbs.gov.au/downloads/2013/07/2013-07-01-extracts.zip" PBS_TF_2013_7.zip, replace
}
foreach i in 2014 2015 2016 {
foreach j in 1 2 3 4 5 6 7 8 9 10 11 12 {
if `j´ < 10 {
copy "https://www.pbs.gov.au/downloads/`i´/0`j´/`i´-0`j´-01-extracts.zip" PBS_TF_`i´_`j´.zip, replac
> e
}
else {
copy "https://www.pbs.gov.au/downloads/`i´/`j´/`i´-`j´-01-extracts.zip" PBS_TF_`i´_`j´.zip, replace
}
}
}

foreach i in 2017 {
foreach j in 1 2 4 5 6 7 8 9  {
copy "https://www.pbs.gov.au/downloads/`i´/0`j´/`i´-0`j´-01-extracts.zip" PBS_TF_`i´_`j´.zip, replac
> e
}
}
foreach i in 2017 {
```

```
foreach j in 10 11 12  {
copy "https://www.pbs.gov.au/downloads/`i´/`j´/`i´-`j´-01-extracts-down-converted.zip" PBS_TF_`i´_`j
> ´.zip, replace
}
}

foreach i in 2018 {
foreach j in 1 2 3 4 5 6 7 8 9 10 11 12 {

if `j´ < 10 {
copy "https://www.pbs.gov.au/downloads/`i´/0`j´/`i´-0`j´-01-extracts-down-converted.zip" PBS_TF_`i´_
> `j´.zip, replace
}
else {
copy "https://www.pbs.gov.au/downloads/`i´/`j´/`i´-`j´-01-extracts-down-converted.zip" PBS_TF_`i´_`j
> ´.zip, replace
}
}
}

local sourcedir "C:\Users\acliv1\Dropbox\~ADAM\PhD\2. Project 2 location and MI outcomes\Stata\PBS D
> ata"
cd "`sourcedir´"

local folder : dir "`sourcedir´" files "*.zip*"

foreach file in `folder´ {
unzipfile `file´, replace
}

*The 2012/12 and 2013/01 data is organised with a folder within the zip folder. So after running the
>  unzip function I had to pull and move the files myself for the following code to work

foreach i in 2010 2011 {
foreach j in 0101 0201 0301 0401 0501 0601 0701 0801 0901 1001 1101 1201 {
import delimited using "drug `i´`j´.txt", delimiters("!") clear
gen year = `i´
gen tmonth = `j´
tostring tmonth, gen(nmonth)
replace nmonth = substr(nmonth,1,length(nmonth)-2)
gen month = real(nmonth)
keep v2 v5 v9 v10 v12 v18 v19 v27 v28 year month
rename (v2 v5 v9 v10 v12 v18 v19 v27 v28) (atc itemcode mq rpts packsize cp2p cdpmq drugname formstr
> ength)
save drug_`i´`j´, replace
}
}

clear
foreach i in 2010 2011 {
foreach j in 0101 0201 0301 0401 0501 0601 0701 0801 0901 1001 1101 1201 {
append using drug_`i´`j´
}
}
save PBS_1, replace

foreach i in 2012 {
foreach j in 0101 0301 0401 0501 0601 0701 0801 0901 1001 {
import delimited using "drug `i´`j´.txt", delimiters("!") clear
gen year = `i´
gen tmonth = `j´
tostring tmonth, gen(nmonth)
replace nmonth = substr(nmonth,1,length(nmonth)-2)
gen month = real(nmonth)
keep v2 v5 v9 v10 v12 v18 v19 v27 v28 year month
rename (v2 v5 v9 v10 v12 v18 v19 v27 v28) (atc itemcode mq rpts packsize cp2p cdpmq drugname formstr
> ength)
save drug_`i´`j´, replace
}
}

import delimited using "drug_20121201.txt", delimiters("!") clear
gen year = 2012
gen month = 12
```

```
keep v2 v5 v9 v10 v12 v18 v19 v27 v28 year month
rename (v2 v5 v9 v10 v12 v18 v19 v27 v28) (atc itemcode mq rpts packsize cp2p cdpmq drugname formstr
> ength)
save drug_20121201, replace

clear
foreach i in 2012 {
foreach j in 0101 0301 0401 0501 0601 0701 0801 0901 1001 1201 {
append using drug_`i´`j´
}
}
save PBS_2, replace

foreach i in 2013 {
foreach j in 0101 0201 0301 0501 0601 0701 0801 0901 1001 1101 1201 {
import delimited using "drug_`i´`j´.txt", delimiters("!") clear
gen year = `i´
gen tmonth = `j´
tostring tmonth, gen(nmonth)
replace nmonth = substr(nmonth,1,length(nmonth)-2)
gen month = real(nmonth)
keep v2 v5 v9 v10 v12 v18 v19 v27 v28 year month
rename (v2 v5 v9 v10 v12 v18 v19 v27 v28) (atc itemcode mq rpts packsize cp2p cdpmq drugname formstr
> ength)
save drug_`i´`j´, replace
}
}
import delimited using "drug_20130403.txt", delimiters("!") clear
gen year = 2013
gen month = 4
keep v2 v5 v9 v10 v12 v18 v19 v27 v28 year month
rename (v2 v5 v9 v10 v12 v18 v19 v27 v28) (atc itemcode mq rpts packsize cp2p cdpmq drugname formstr
> ength)
save drug_20130401, replace

clear
foreach i in 2013 {
foreach j in 0101 0201 0301 0401 0501 0601 0701 0801 0901 1001 1101 1201 {
append using drug_`i´`j´
}
}
save PBS_3, replace

foreach i in 2014 2015 2016 {
foreach j in 0101 0201 0301 0401 0501 0601 0701 0801 0901 1001 1101 1201 {
import delimited using "drug_`i´`j´.txt", delimiters("!") clear
gen year = `i´
gen tmonth = `j´
tostring tmonth, gen(nmonth)
replace nmonth = substr(nmonth,1,length(nmonth)-2)
gen month = real(nmonth)
keep v2 v5 v9 v10 v12 v18 v19 v27 v28 year month
rename (v2 v5 v9 v10 v12 v18 v19 v27 v28) (atc itemcode mq rpts packsize cp2p cdpmq drugname formstr
> ength)
save drug_`i´`j´, replace
}
}

clear
foreach i in 2014 2015 2016 {
foreach j in 0101 0201 0301 0401 0501 0601 0701 0801 0901 1001 1101 1201 {
append using drug_`i´`j´
}
}
save PBS_4, replace

foreach i in 2017 {
foreach j in 0101 0201 0401 0501 0601 0701 0801 1001 1101 1201 {
import delimited using "drug_`i´`j´.txt", delimiters("!") clear
gen year = `i´
gen tmonth = `j´
```

```
tostring tmonth, gen(nmonth)
replace nmonth = substr(nmonth,1,length(nmonth)-2)
gen month = real(nmonth)
keep v2 v5 v9 v10 v12 v18 v19 v27 v28 year month
rename (v2 v5 v9 v10 v12 v18 v19 v27 v28) (atc itemcode mq rpts packsize cp2p cdpmq drugname formstr
> engthh)
save drug_`i´`j´, replace
}
}
import delimited using "drug_20170901.txt", delimiters("!") clear
gen year = 2017
gen month = 9
rename (pksz commp2p commdpmq) (packsize cp2p cdpmq)
keep atc itemcode mq rpts packsize cp2p cdpmq drugname formstrength year month
save drug_20170901, replace
br
foreach i in 2017 {
foreach j in 0101 0201 0401 0501 0601 0701 0801 0901 1001 1101 1201 {
append using drug_`i´`j´
}
}
save PBS_5, replace
foreach i in 2018 {
foreach j in 0101 0201 0301 0401 0501 0601 0701 0801 0901 1001 1101 1201 {
import delimited using "drug_`i´`j´.txt", delimiters("!") clear
gen year = `i´
gen tmonth = `j´
tostring tmonth, gen(nmonth)
replace nmonth = substr(nmonth,1,length(nmonth)-2)
gen month = real(nmonth)
keep v2 v5 v9 v10 v12 v18 v19 v27 v28 year month
rename (v2 v5 v9 v10 v12 v18 v19 v27 v28) (atc itemcode mq rpts packsize cp2p cdpmq drugname formstr
> engthh)
save drug_`i´`j´, replace
}
}
foreach i in 2018 {
foreach j in 0101 0201 0301 0401 0501 0601 0701 0801 0901 1001 1101 1201 {
append using drug_`i´`j´
}
}
save PBS_6, replace
clear
forval i=1/6 {
append using PBS_`i´
}
rename (mq rpts) (medication_quantity repeats)
save PBS_ALL, replace
```

There were no changes in packsize across medications based on their respective itemcodes, so a new reference set with only one observation per itemcode and packsize was created.

```
use "G:\Adam\Resources and codes\PBS_ALL", clear
br
bysort itemcode packsize : gen nghost=_n
bysort itemcode packsize : gen diff = packsize - packsize[n-1]
ta diff
bysort itemcode :keep if _n ==1
gen pbs_item_code = substr("000000", 1, 6-length(itemcode)) + itemcode
keep pbs_item_code packsize
br
save PBS_packsizes, replace
```

5.6 Separate dataset into drug classes for PDC calculation

Using the dataset for people alive at 90 days post MI, we split the dataset into 8 sets, once for each class (1-7) and combination of ACEi and ARB (8). The datasets will be used for PDC calculations so that drugs are analysed for PDC in their respective class. This also allowed for analysis of drug dispensing within 90 days for each medication class. As aspirin is primiarly purchased as an over the counter item rathern through the PBS, this dataset was not analysed.
We were also able to use this dataset and merge it with the MI cohort to identify how many people received none, one, two, three, or four classes of medications within 90 days of discharge.

```
use MI_drugs_alive90d, clear
drop if supplydate >= sepdate + 365
br

gen drug_class=0
replace drug_class = 1 if P2Y12==1
replace drug_class = 3 if beta==1
replace drug_class = 4 if statin==1
replace drug_class = 5 if aspirin ==1
replace drug_class = 6 if ace ==1
replace drug_class = 7 if arb ==1
ta drug_class
save MI_drugs_class, replace

forval i = 1/7 {
use MI_drugs_class, clear
keep if drug_class == `i´
keep ppn admdate sepdate deathdate supplydate pbs_item_code drug_class
save MI_drugs_class_`i´, replace
}
use MI_drugs_class, clear
keep if ace_arb == 1
keep ppn admdate sepdate deathdate supplydate pbs_item_code drug_class
save MI_drugs_class_2, replace
}

use MI_drugs_class, clear
collapse (sum) beta_90d ace_90d arb_90d P2Y12_90d statin_90d ace_arb_90d, by(ppn admdate sepdate)
foreach i in beta_90d ace_90d arb_90d P2Y12_90d statin_90d ace_arb_90d {
replace `i´ = 1  if `i´ >= 1
}
save MI_drugs_received, replace

use MI_cohort_NDI, clear
keep if inrange(sepdate,td(1/7/2012),td(30/6/2017))
merge 1:1 ppn admdate using MI_drugs_received
br if deathdate > sepdate + 90 & _merge != 3
keep ppn admdate beta_90d-ace_arb_90d
gen OMT_score = beta_90d + P2Y12_90d + statin_90d + ace_arb_90d
replace OMT_score = 0 if OMT_score == .
ta OMT_score

save MI_drugs_received_score, replace
```

5.7 Preadmission use of medications

In order to describe medication use prior to MI admission, a dataset was created that shows the proportion of MI admissions that received sceondary prevention medications (by class) in the 90 days prior to admission

```
use MI_drugs_all, clear
br
keep if supplydate > admdate - 90
keep if supplydate < admdate
keep ppn admdate sepdate priorMI supplydate beta-P2Y12
drop aspirin
gen dist = admdate - supplydate

collapse (sum) beta ace arb statin ace_arb P2Y12, by (ppn admdate sepdate priorMI)
```

```
foreach i in beta ace arb statin ace_arb P2Y12 {
replace `i´ = 1 if `i´ >= 1
}
ta beta if priorMI == 0
ta beta if priorMI == 1
ta ace_arb if priorMI == 0
ta ace_arb if priorMI == 1
ta statin if priorMI == 0
ta statin if priorMI == 1
ta P2Y12 if priorMI == 0
ta P2Y12 if priorMI == 1
rename (beta ace arb statin ace_arb P2Y12) (prior_beta prior_ace prior_arb prior_statin prior_ace_ar
> b prior_P2Y12)
save MI_cohort_preMIdrugs, replace
```

5.8 PDC calculation method

A five step process was undertaken to determine both the PDC values according to a 365 day observation window following discharge. This method was adapted from previous reported methods from a Stata module design [2].

- Step 1: The data was reshaped into wide form, and days covered by the supply generated according to the defined daily dose from WHO or via the corrected method used for beta blockers.
- Step 2:End dates for follow up were created, with date of death replacing the end of follow up date if the indivudal died sooner than 365 days following discharge.
- Step 3: Adjustments were made to supply dates if there was overlap from the previous supply and the next supply based on the number of days covered by the respective pack size.
- Step 4: Values for the 365 days of follow up were created and using the adjusted supply dates, it could be determined whether the individual had access to the medication or not. We then adjusted days covered should supply continue beyond the recorded day of death.
- Step 5: PDC was caluclated using both a 365 window as well as the adjusted window of 365 post discharge or days until death, which came first.

5.8.1 PDC calculation- statins

```
*Step 1: Reshape data
use MI_drugs_class_4, clear
drop if supplydate <= admdate - 60
bysort ppn admdate (supplydate): gen nghost=_n
ta nghost
br
gen dayssupply=30
drop pbs_item_code
count if supplydate < sepdate
gen supplygap = 0
replace supplygap = admdate - supplydate if supplydate < admdate
replace supplydate= sepdate - supplygap if supplygap < dayssupply & supplygap > 0
drop supplygap
reshape wide supplydate dayssupply, i(ppn admdate) j(nghost)
}
*Step 2: Create enddates
```

```
{
gen enddate = sepdate +365
format enddate %td
replace enddate = deathdate if deathdate < enddate
gen study_days = enddate - sepdate
count if study_days <90
}
*Step 3: Adjust for when next supply overlaps with previous supply.
{
forval i = 2/52 {
local j = `i´ - 1
replace supplydate`i´= (supplydate`j´ + dayssupply`j´) if (supplydate`i´ < (supplydate`j´ + dayssupp
> ly`j´) & supplydate`i´ !=.)
}
}
*Step 4: Generate values for the 365 days of the year.
{
quietly {
forval i = 1/365 {
gen day`i´ = 0
forval j = 1/52 {
replace day`i´ = 1 if supplydate`j´ <= sepdate + `i´ & sepdate + `i´ <= supplydate`j´ + dayssupply`j
> ´ & supplydate`j´ !=.
}
}
}
}
*Removes counts for prescriptions supplied after deathdate/end of study follow up
forval i = 1/365 {
replace day`i´ = 0 if `i´ > study_days
}
*Step 5: Calculate PDC
egen days_covered = rsum(day1-day365)
gen PDC_365_statins = days_covered / 365
gen PDC_adj_statins = days_covered/ study_days
count if PDC_365_statins == 0
save MI_cohort_PDC_wide_statins, replace
```

5.8.2 PDC calculation- anti-platelets

```
*Step 1: Reshape data
{
use MI_drugs_class_1, clear
drop if supplydate <= sepdate - 60
merge m:1 pbs_item_code using PBS_packsizes
keep if _merge==3
drop _merge
replace packsize = 28 if packsize == 56
ta packsize
br
bysort ppn admdate (supplydate): gen nghost=_n
ta nghost
rename packsize dayssupply
drop pbs_item_code
gen supplygap = 0
replace supplygap = admdate - supplydate if supplydate < admdate
replace supplydate= sepdate - supplygap if supplygap < dayssupply & supplygap > 0
drop supplygap
reshape wide supplydate dayssupply, i(ppn admdate) j(nghost)
}
*Step 2: Create enddates
{
gen enddate = sepdate +365
format enddate %td
replace enddate = deathdate if deathdate < enddate
```

```
gen study_days = enddate - sepdate
count if study_days <90
}
*Step 3: Adjust for when next supply overlaps with previous supply.
{
forval i = 2/30 {
local j = `i´ - 1
replace supplydate`i´= (supplydate`j´ + dayssupply`j´) if (supplydate`i´ < (supplydate`j´ + dayssupp
> ly`j´) & supplydate`i´ !=.)
}
}
*Step 4: Generate values for the 365 days of the year.
{
quietly {
forval i = 1/365 {
gen day`i´ = 0
forval j = 1/30 {
replace day`i´ = 1 if supplydate`j´ <= sepdate + `i´ & sepdate + `i´ <= supplydate`j´ + dayssupply`j
> ´ & supplydate`j´ !=.
}
}
}
}
*Removes counts for prescriptions supplied after deathdate/end of study follow up
forval i = 1/365 {
replace day`i´ = 0 if `i´ > study_days
}
*Step 5: Calculate PDC by adding the row values together from day 1 to 365 and dividing by 365, then
>  create an adjusted figure that considers days avaiable based on sepdate to enddate, where enddate
>  is truncated to deathdate
egen days_covered = rsum(day1-day365)
gen PDC_365_P2Y12 = days_covered / 365
gen PDC_adj_P2Y12 = days_covered/ study_days
count if PDC_365_P2Y12 == 0
save MI_cohort_PDC_wide_P2Y12, replace
```

5.8.3 PDC calculation- ACEi

```
use MI_drugs_class_6, clear
drop if supplydate <= sepdate - 60
merge m:1 pbs_item_code using PBS_packsizes
keep if _merge==3
drop _merge
replace packsize = 30 if packsize ==90
replace packsize = 38 if packsize ==1
replace packsize = 28 if packsize == 56
ta packsize
br
bysort ppn admdate (supplydate): gen nghost=_n
ta nghost
rename packsize dayssupply
drop pbs_item_code
gen supplygap = 0
replace supplygap = admdate - supplydate if supplydate < admdate
replace supplydate= sepdate - supplygap if supplygap < dayssupply & supplygap > 0
drop supplygap
reshape wide supplydate dayssupply, i(ppn admdate) j(nghost)
}
*Step 2: Create enddates
{
gen enddate = sepdate +365
format enddate %td
replace enddate = deathdate if deathdate < enddate
gen study_days = enddate - sepdate
count if study_days <90
```

```
}
*Step 3: Adjust for when next supply overlaps with previous supply.
{
forval i = 2/34 {
local j = `i´ - 1
replace supplydate`i´= (supplydate`j´ + dayssupply`j´) if (supplydate`i´ < (supplydate`j´ + dayssupp
> ly`j´) & supplydate`i´ !=.)
}
}
*Step 4: Generate values for the 365 days of the year.
{
quietly {
forval i = 1/365 {
gen day`i´ = 0
forval j = 1/34 {
replace day`i´ = 1 if supplydate`j´ <= sepdate + `i´ & sepdate + `i´ <= supplydate`j´ + dayssupply`j
> ´ & supplydate`j´ !=.
}
}
}
}
*Removes counts for prescriptions supplied after deathdate/end of study follow up
forval i = 1/365 {
replace day`i´ = 0 if `i´ > study_days
}
*Step 5: Calculate PDC
egen days_covered = rsum(day1-day365)
gen PDC_365_ACE = days_covered / 365
gen PDC_adj_ACE = days_covered/ study_days
count if PDC_365_ACE == 0
save MI_cohort_PDC_wide_ACE, replace
```

5.8.4 PDC calculation- ARB

```
*Step 1: Reshape data
{
use MI_drugs_class_7, clear
drop if supplydate <= sepdate - 60
merge m:1 pbs_item_code using PBS_packsizes
keep if _merge==3
drop _merge
replace packsize = 30 if packsize ==90
replace packsize = 28 if packsize == 56
ta packsize
bysort ppn admdate (supplydate): gen nghost=_n
ta nghost
rename packsize dayssupply
drop pbs_item_code
gen supplygap = 0
replace supplygap = admdate - supplydate if supplydate < admdate
replace supplydate= sepdate - supplygap if supplygap < dayssupply & supplygap > 0
drop supplygap
reshape wide supplydate dayssupply, i(ppn admdate) j(nghost)
}
*Step 2: Create enddates
{
gen enddate = sepdate +365
format enddate %td
replace enddate = deathdate if deathdate < enddate
gen study_days = enddate - sepdate
count if study_days <90
}
*Step 3: Adjust for when next supply overlaps with previous supply.
{
forval i = 2/35 {
```

```
local j = `i´ - 1
replace supplydate`i´= (supplydate`j´ + dayssupply`j´) if (supplydate`i´ < (supplydate`j´ + dayssupp
> ly`j´) & supplydate`i´ !=.)
}
}
*Step 4: Generate values for the 365 days of the year.
{
quietly {
forval i = 1/365 {
gen day`i´ = 0
forval j = 1/35 {
replace day`i´ = 1 if supplydate`j´ <= sepdate + `i´ & sepdate + `i´ <= supplydate`j´ + dayssupply`j
> ´ & supplydate`j´ !=.
}
}
}
}
*Removes counts for prescriptions supplied after deathdate/end of study follow up
{
forval i = 1/365 {
replace day`i´ = 0 if `i´ > study_days
}
*Step 5: Calculate PDC
egen days_covered = rsum(day1-day365)
gen PDC_365_ARB = days_covered / 365
gen PDC_adj_ARB = days_covered/ study_days
count if PDC_365_ARB == 0
save MI_cohort_PDC_wide_ARB, replace
}
```

5.8.5 PDC calculation- ACEi or ARB

Both datasets for ACEI and ARB were appended and then collapsed with respect to MI admission, with double ups removed and assuming the patient stopped one treatment and commenced the new treatment, similar to how it was assumed with drugs within the same classes for the other datasets.

```
use MI_cohort_PDC_wide_ACE, clear
append using MI_cohort_PDC_wide_ARB
sort ppn admdate
bysort ppn admdate : gen nghost = _n
ta nghost
collapse day1-day365, by(ppn admdate study_days)
forval i = 1/365 {
replace day`i´ = 1 if day`i´ > 0
}
egen days_covered = rsum(day1-day365)
gen PDC_365_ACE_ARB = days_covered / 365
gen PDC_adj_ACE_ARB = days_covered/ study_days
save MI_cohort_PDC_wide_ACE_ARB, replace
```

5.8.6 PDC calculation- Beta blockers

Due to variance in defined daily dosing regimens within the beta blocker group, the 75% rule was incorporated [3]. This involved taking the 75th percentile of refill times to determine days supply per PBS item number. This value was then applied across all beta blocker drugs defined by ATC code.

```
use MI_drugs_class_3, clear
br
ta pbs_item_code
egen betagroup = group(pbs_item_code)
ta betagroup
bysort ppn admdate pbs_item_code (supplydate) : gen nghost= _n
```

```
bysort ppn admdate pbs_item_code (supplydate): gen fill_diff = supplydate[_n+1] - supplydate
bysort betagroup : egen p75  = pctile(fill_diff), p(75)
drop nghost fill_diff
save MI_drugs_class_3_packsize, replace

*Step 1: Reshape data
{
use MI_drugs_class_3_packsize, clear
drop if supplydate <= sepdate - 60
bysort ppn admdate (supplydate): gen nghost=_n
ta nghost
rename p75 dayssupply
drop pbs_item_code betagroup
gen supplygap = 0
replace supplygap = admdate - supplydate if supplydate < admdate
replace supplydate= sepdate - supplygap if supplygap < dayssupply & supplygap > 0
drop supplygap
reshape wide supplydate dayssupply, i(ppn admdate) j(nghost)
}
*Step 2: Create enddates
{
gen enddate = sepdate +365
format enddate %td

replace enddate = deathdate if deathdate < enddate
gen study_days = enddate - sepdate

count if study_days <90
}
*Step 3: Adjust for when next supply overlaps with previous supply.
{
forval i = 2/35 {
local j = `i´ - 1
replace supplydate`i´= (supplydate`j´ + dayssupply`j´) if (supplydate`i´ < (supplydate`j´ + dayssupp
> ly`j´) & supplydate`i´ !=.)
}
}
*Step 4: Generate values for the 365 days of the year.
{
quietly {
forval i = 1/365 {
gen day`i´ = 0
forval j = 1/35 {
replace day`i´ = 1 if supplydate`j´ <= sepdate + `i´ & sepdate + `i´ <= supplydate`j´ + dayssupply`j
> ´ & supplydate`j´ !=.
}
}
}
}
*Removes counts for prescriptions supplied after deathdate/end of study follow up
forval i = 1/365 {
replace day`i´ = 0 if `i´ > study_days
}
*Step 5: Calculate PDC
egen days_covered = rsum(day1-day365)
gen PDC_365_beta = days_covered / 365
gen PDC_adj_beta = days_covered/ study_days
count if PDC_365_beta == 0
save MI_cohort_PDC_wide_beta, replace
```

5.9 Linking postcode with SEIFA

A combination of the PBS and MBS datasets were used to determine the postcode of the indivudal which is needed to match to socioeconomic disadvantage using a reference dataset of postcodes and IRSD scores.
We first looked the MBS set which lists dates of service events and the postcode recorded by MBS at the time of the event. We merged this with the cohort to find the most recent service event that occurred prior to the MI.

```
*Generate last event to only keep services before then
{
use MI_cohort_NDI, clear
keep if sepdate >= td(1,7,2012)
bysort ppn (sepdate) : keep if _n == _N
keep ppn sepdate
save lastMI, replace
}
*All services before last sepdate
{
forval i = 0/204 {
use MBS_`i´, clear
merge m:1 ppn using lastMI
keep if _merge == 3
drop _merge
keep if dos <= sepdate
save MBS_`i´last, replace
}
clear
forval i = 0/204 {
append using MBS_`i´last
}
gen servid = _n
save MBSlast, replace
}
*Merge in MI cohort and MBS service events
{
use MI_cohort_NDI, clear
keep if sepdate >= td(1,7,2012)
bysort ppn (sepdate) : gen nghost = _n
ta nghost
forval i = 2(1)12 {
use MI_cohort_NDI, clear
keep if sepdate >= td(1,7,2012)
bysort ppn (sepdate) : keep if _n == `i´
merge 1:m ppn using MBSlast
keep if _merge == 3
drop _merge
gen dist = sepdate-dos
drop if dist < 0
bysort ppn admdate (dist) : keep if _n == 1
keep ppn admdate patient_post
save MIPCLB_`i´, replace
}
use MI_cohort_NDI, clear
keep if sepdate >= td(1,7,2012)
bysort ppn (sepdate) : keep if _n == 1
merge 1:m ppn using MBSlast
keep if _merge == 3
drop _merge
gen dist = sepdate-dos
drop if dist < 0
bysort ppn admdate (dist) : keep if _n == 1
keep ppn admdate patient_post
forval i = 2/12 {
append using MIPCLB_`i´
}
save MIPCLB, replace
}
{
use MI_cohort_NDI, clear
keep if sepdate >= td(1,7,2012)
merge 1:1 ppn admdate using MIPCLB
keep if _merge == 1
drop _merge
```

```
drop patient
save PCMIS, replace
```

As there were 3,006 MI admissions without a service event preceeding the MI, the same process was applied but looking at service events that occured after the MI. The most recent service event's associated postcode was selected in the event of multiple events occuring following the MI.

```
forval i = 2(1)12 {
use PCMIS, clear
bysort ppn (sepdate) : keep if _n == `i´
merge 1:m ppn using MBSlast
keep if _merge == 3
drop _merge
gen dist = dos-sepdate
bysort ppn admdate (dist) : keep if _n == 1
keep ppn admdate patient_post
save MIPCLB2_`i´, replace
}
use PCMIS, clear
bysort ppn (sepdate) : keep if _n == 1
merge 1:m ppn using MBSlast
keep if _merge == 3
drop _merge
gen dist = dos-sepdate
bysort ppn admdate (dist) : keep if _n == 1
keep ppn admdate patient_post
forval i = 2/12 {
append using MIPCLB2_`i´
}
save MIPCLB2, replace
use MIPCLB, clear
append using MIPCLB2
save MIPCLB_forward_back, replace
use MI_cohort_NDI, clear
keep if inrange(sepdate,td(1/7/2012),td(30/6/2017))
merge 1:1 ppn admdate using MIPCLB
keep if _merge == 1
drop _merge
save MI_cohort_nopostcode, replace
```

This still left 2,477 MI admissions from the cohort without a postcode avaiable for matching with IRSD scores. The PBS dataset also contains postcode information, and so the same process was applied but using the PBS dispensing date closest to the index MI admission.

```
foreach i in A B C D G H J L M N P R S V Z {
use MI_cohort_nopostcode, clear
drop patient_postcode
bysort ppn (sepdate) : keep if _n == 1
merge 1:m ppn using "G:/Jed/PBS `i´.dta"
keep if _merge == 3
drop _merge
save PCMIS_PBS_`i´, replace
}
clear
foreach i in A B C D G H J L M N P R S V Z {
append using PCMIS_PBS_`i´
}
gen dos = date(date_of_s,"DMY")
format dos %td
gen dist = dos-sepdate
bysort ppn admdate (dist) : keep if _n == 1
keep ppn admdate patient_post
append using MIPCLB_forward_back
bysort ppn admdate: keep if _n == 1
```

```
save MIPCLB_forward_back_PBS, replace

use MI_cohort_NDI, clear
keep if inrange(sepdate,td(1/7/2012),td(30/6/2017))
merge 1:1 ppn admdate using MIPCLB_forward_back_PBS
keep if _merge == 1
keep ppn admdate patient_
save MI_cohort_nopostcode_MBS_PBS, replace

use MI_drugs_received, clear
merge 1:1 ppn admdate using MI_cohort_nopostcode_MBS_PBS
```

This left 2093 MI admissions without a postcode, and after merging with the MI cohort for analysis, only 59 MI admissions were left without a postcode and therefore could not be assigned an IRSD score.
The next step involved merging the postcode data with the reference dataset.

```
use MIPCLB_forward_back_PBS, clear
rename patient_postcode postcode
merge m:1 postcode using "G:\Jed\Costing studies\Data\SESPC"
*580 postcodes not matched, so next nearest postcode assigned and matched
replace postcode = postcode - 1 if _merge == 1
drop if _merge == 2
drop _merge IRSD quint
merge m:1 postcode using "G:\Jed\Costing studies\Data\SESPC"

drop if _merge == 2
drop _merge
bysort ppn admdate : gen nghost = _n
ta nghost
drop nghost
save MIPCLB_SES, replace

use MI_cohort_NDI, clear
keep if inrange(sepdate,td(1/7/2012),td(30/6/2017))
merge 1:1 ppn admdate using MIPCLB_SES
drop if _merge == 2
drop _merge

save MI_cohort_SES_withblanks, replace
```

As there were 59 MI admissions without IRSD scores, this set were assigned the mean IRSD score from the rest of the cohort.

```
use MI_cohort_SES_withblanks, clear
egen IRSD_score_mean = mean(IRSD_score)
replace IRSD_score = IRSD_score_mean if IRSD_score ==.
count if IRSD_score == .
save MI_cohort_SES, replace
```

6 Analysis

6.1 Creation of analysis data set

Each of the separate datasets throughout MI cohort creation, remoteness score allocation, comorbidity allocation, treatment outcome allocation, medication dispensing, medication adherence, and socioeconomic disadvtage score allocation were merged into one dataset.
In order to ensure there were dispensing and PDC scores for MI admissions where there were no PBS data were defined to have not receieved therapy and not adherent to therapy. These admissions replaced blank values following the merge with 0 values to ensure they could be anlysed in the regression model.
Lastly, MIs where no ARIA scores were available were dropped from the analysis dataset.

```
use MI_cohort_ndi, clear
br
keep if inrange(sepdate,td(1/7/2012),td(30/6/2017))
gen death_time = deathdate-sepdate
```

```
gen hosp_mort = 0
replace hosp_mort = 1 if death_time < 1
replace hosp_mort = 2 if death_time >= 1 & death_time <=90
ta hosp_mort
drop death_time
drop sepmode
save MI_ADS0, replace

use MI_cohort_PDC_wide_beta, clear
keep ppn admdate PDC_365_beta PDC_adj_beta lastmonthbeta
gen beta80adj = 1
replace beta80adj = 0 if PDC_adj_beta <0.8
ta beta80adj
br
save MI_ADS1, replace

use MI_cohort_PDC_wide_statins, clear
keep ppn admdate PDC_365_statins PDC_adj_statins lastmonthstatin
br
gen statin80adj = 1
replace statin80adj = 0 if PDC_adj_statin <0.8
ta statin80adj
save MI_ADS2, replace

use MI_cohort_PDC_wide_P2Y12, clear
keep ppn admdate PDC_365_P2Y12 PDC_adj_P2Y12 lastmonthP2Y12
br
gen P2Y1280adj = 1
replace P2Y1280adj = 0 if PDC_adj_P2Y12 <0.8
ta P2Y1280adj
save MI_ADS3, replace

use MI_cohort_PDC_wide_ACE, clear
keep ppn admdate PDC_365_ACE PDC_adj_ACE lastmonthACE
br
save MI_ADS4, replace

use MI_cohort_PDC_wide_ARB, clear
keep ppn admdate PDC_365_ARB PDC_adj_ARB lastmonthARB
br
save MI_ADS5, replace

use MI_cohort_PDC_wide_ACE_ARB, clear
keep ppn admdate PDC_365_ACE_ARB PDC_adj_ACE_ARB lastmonthACE_ARB
br
gen ACE_ARB80adj = 1
replace ACE_ARB80adj = 0 if PDC_adj_ACE_ARB <0.8
ta ACE_ARB80adj
save MI_ADS6, replace

use MI_drugs_received_score, clear
br
save MI_ADS7, replace

use MI_sla_match_VIC_noblanks, clear
keep ppn admdate ARIA Mean
br
rename Mean mean_ARIA
save MI_ADS8, replace

use MI_cohort_SES, clear
keep ppn admdate IRSD_score quint postcode
save MI_ADS9, replace

use MI_cohort_preMIdrugs, clear
keep ppn admdate prior_beta-prior_P2Y12
save MI_ADS10, replace

use MI_comorbid, clear
bysort ppn admdate : keep if _n==1
save MI_ADS11, replace

use MI_proc_clean, clear
drop sepdate
save MI_ADS12, replace

use MI_ADS0, clear
```

```
br
forval i = 1/12 {
merge 1:1 ppn admdate using MI_ADS`i´
drop if _merge == 2
drop _merge
}
br
replace agegroup = substr(agegroup,1,2)
destring agegroup, replace force
ta agegroup
gen dead90 = 0
replace dead90 = 1 if deathdate <= sepdate + 90
gen dead365 = 0
replace dead365 = 1 if deathdate <= sepdate + 365
*Remove MIs with no ARIA
count if mean_ARIA == .
drop if mean_ARIA == .
*replace dispensing with 0 if alive at 90 days
foreach i in beta_90d ace_90d arb_90d ace_arb_90d P2Y12_90d statin_90d {
replace `i´ = 0 if hosp_mort == 0 & `i´ ==.
}
save MI_ADS_ALL_SA, replace
*replace PDC_adj values with 0 if alive but no dispensing
foreach i in PDC_adj_beta PDC_adj_statins PDC_adj_P2Y12 PDC_adj_ACE PDC_adj_ARB PDC_adj_ACE_ARB PDC_
> adj_beta {
replace `i´ = 0 if hosp_mort == 0 & `i´ ==.
}
save MI_ADS_ALL, replace
}
```

6.2 Result tables

6.2.1 Tables of total population characteristics

This first table uses the total MI cohort within Victoria, and so it includes admissions where people died during admission or within 90 days of discharge. This table lists total population characteristics, as well as characteristics by STEMI or NSTEMI, as well as characteristics uses categories of remoteness (HA, A, MA).

```
use MI_ADS_ALL, clear
forval i = 35(10)85 {
replace agegroup = agegroup - 5 if agegroup == `i´
}
save MI_ADS_ALL_age10, replace
}
*Create STEMI and NSTEMI table
{
use MI_ADS_ALL_age10, clear
br
gen tahelp = 1

ta tahelp STEMI, matcell(B1)
ta sex STEMI, matcell(B2)
ta agegroup STEMI, matcell(B3)
ta HT STEMI if HT==1, matcell(B5)
ta AF STEMI if AF==1, matcell (B6)
ta DM STEMI if DM==1, matcell(B7)
ta HF STEMI if HF==1, matcell (B8)
ta IS STEMI if IS==1, matcell (B9)
ta PCI_tag STEMI if PCI_tag==1, matcell (B10)
ta CABG_tag STEMI if CABG_tag==1, matcell (B11)
ta hosp_mort STEMI if hosp_mort == 1, matcell(B24)
ta hosp_mort STEMI if hosp_mort == 2, matcell(B25)
ta prior_P2Y12 STEMI if prior_P2Y12 == 1 , matcell (B12)
ta prior_statin STEMI if prior_statin == 1, matcell (B13)
ta prior_ace STEMI if prior_ace == 1, matcell (B14)
```

```
ta prior_arb STEMI if prior_arb == 1, matcell (B15)
ta prior_beta STEMI if prior_beta == 1, matcell (B16)
ta P2Y12_90d STEMI if P2Y12_90d==1 , matcell (B17)
ta statin_90d STEMI if statin_90d==1, matcell (B18)
ta ace_90d STEMI if ace_90d==1, matcell (B19)
ta arb_90d STEMI if arb_90d==1, matcell (B20)
ta ace_arb_90d STEMI if ace_arb_90d == 1, matcell(B23)
ta beta_90d STEMI if beta_90d==1, matcell (B21)
ta OMT_score STEMI, matcell(B22)

matrix B = (B1\B2\B3\B5\B6\B7\B8\B9\B10\B11\B24\B25\B12\B13\B14\B15\B16\B17\B18\B19\B20\B23\B21\B22)
matrix list B
clear
svmat B
br

rename (B1 B2) (NSTEMI STEMI)
egen total = max(NSTEMI + STEMI)
gen totalproportion = string((100 * (STEMI + NSTEMI) / total), "%3.0f")+"%"
replace total = NSTEMI + STEMI

foreach i in STEMI NSTEMI {
egen `i´total = max(`i´)
gen `i´proportion1 = (100 * `i´ / `i´total)
replace `i´proportion1 = `i´total / total * 100 if `i´proportion1 == 100
gen `i´proportion = string(`i´proportion1, "%3.0f")+"%"
drop `i´proportion1
}

gen id = _n
gen demoname = ""
replace demoname = "Male" if _n== 2
replace demoname = "Female" if _n== 3
replace demoname = "Aged 30-39" if _n== 4
replace demoname = "Aged 40-49" if _n== 5
replace demoname = "Aged 50-54" if _n== 6
replace demoname = "Aged 60-64" if _n== 7
replace demoname = "Aged 70-74" if _n== 8
replace demoname = "Aged 80+" if _n== 9
replace demoname = "Hypertension" if _n== 10
replace demoname = "Atrial fibrillation" if _n== 11
replace demoname = "Diabetes mellitus" if _n== 12
replace demoname = "Heart failure" if _n== 13
replace demoname = "Ischaemic stroke" if _n== 14
replace demoname = "PCI within 90 days of MI" if _n== 15
replace demoname = "CABG within 90 days of MI" if _n== 16
replace demoname = "Died during admission" if _n== 17
replace demoname = "Died within 90 days of discharge" if _n== 18
replace demoname = "prior P2Y12i" if _n == 19
replace demoname = "prior Statin" if _n == 20
replace demoname = "prior ACEi" if _n == 21
replace demoname = "prior ARB" if _n == 22
replace demoname = "prior Beta blocker" if _n == 23
replace demoname = "P2Y12i" if _n == 24
replace demoname = "Statin" if _n == 25
replace demoname = "ACEi" if _n == 26
replace demoname = "ARB" if _n == 27
replace demoname = "ACEi or ARB" if _n == 28
replace demoname = "Beta blocker" if _n == 29
replace demoname = "No medication dispensed"  if _n == 30
replace demoname = "One medication class dispensed"  if _n == 31
replace demoname = "Two medication classes dispensed" if _n == 32
replace demoname = "Three medication classes dispensed"  if _n == 33
replace demoname = "Four medication classes dispensed" if _n == 34

drop NSTEMItotal STEMItotal
order demoname total totalproportion NSTEMI NSTEMIproportion STEMI STEMIproportion
save totalpopACS, replace
}
*Create table with ARIA
{
```

```
use MI_ADS_ALL_age10, clear
br
gen tahelp = 1

ta tahelp ARIA, matcell(B1)
ta sex ARIA, matcell(B2)
ta agegroup ARIA, matcell(B3)
ta HT ARIA if HT==1, matcell(B5)
ta AF ARIA if AF==1, matcell (B6)
ta DM ARIA if DM==1, matcell(B7)
ta HF ARIA if HF==1, matcell (B8)
ta IS ARIA if IS==1, matcell (B9)
ta PCI_tag ARIA if PCI_tag==1, matcell (B10)
ta CABG_tag ARIA if CABG_tag==1, matcell (B11)
ta hosp_mort ARIA if hosp_mort == 1, matcell(B24)
ta hosp_mort ARIA if hosp_mort == 2, matcell(B25)
ta prior_P2Y12 ARIA if prior_P2Y12 == 1 , matcell (B12)
ta prior_statin ARIA if prior_statin == 1, matcell (B13)
ta prior_ace ARIA if prior_ace == 1, matcell (B14)
ta prior_arb ARIA if prior_arb == 1, matcell (B15)
ta prior_beta ARIA if prior_beta == 1, matcell (B16)
ta P2Y12_90d ARIA if P2Y12_90d==1 , matcell (B17)
ta statin_90d ARIA if statin_90d==1, matcell (B18)
ta ace_90d ARIA if ace_90d==1, matcell (B19)
ta arb_90d ARIA if arb_90d==1, matcell (B20)
ta ace_arb_90d ARIA if ace_arb_90d == 1, matcell(B23)
ta beta_90d ARIA if beta_90d==1, matcell (B21)
ta OMT_score ARIA, matcell(B22)

matrix analpoparia = (B1\B2\B3\B5\B6\B7\B8\B9\B10\B11\B24\B25\B12\B13\B14\B15\B16\B17\B18\B19\B20\B2
> 3\B21\B22)
matrix list analpoparia
clear
svmat analpoparia
br

rename (analpoparia1 analpoparia2 analpoparia3) (A HA MA)
order HA A MA
egen total = max(HA+A+MA)
gen totalproportion = string((100 * (HA+A+MA) / total), "%3.0f")+"%"
replace total = HA+A+MA

foreach i in HA A MA {
egen `i´total = max(`i´)
gen `i´proportion1 = (100 * `i´ / `i´total)
replace `i´proportion1 = `i´total / total * 100 if `i´proportion1 == 100
gen `i´proportion = string(`i´proportion1, "%3.0f")+"%"
drop `i´proportion1
}

gen id = _n
gen demoname = ""
replace demoname = "Male" if _n== 2
replace demoname = "Female" if _n== 3
replace demoname = "Aged 30-39" if _n== 4
replace demoname = "Aged 40-49" if _n== 5
replace demoname = "Aged 50-54" if _n== 6
replace demoname = "Aged 60-64" if _n== 7
replace demoname = "Aged 70-74" if _n== 8
replace demoname = "Aged 80+" if _n== 9
replace demoname = "Hypertension" if _n== 10
replace demoname = "Atrial fibrillation" if _n== 11
replace demoname = "Diabetes mellitus" if _n== 12
replace demoname = "Heart failure" if _n== 13
replace demoname = "Ischaemic stroke" if _n== 14
replace demoname = "PCI within 90 days of MI" if _n== 15
replace demoname = "CABG within 90 days of MI" if _n== 16
replace demoname = "Died during admission" if _n== 17
replace demoname = "Died within 90 days of discharge" if _n== 18
replace demoname = "prior P2Y12i" if _n == 19
replace demoname = "prior Statin" if _n == 20
```

```
replace demoname = "prior ACEi" if _n == 21
replace demoname = "prior ARB" if _n == 22
replace demoname = "prior Beta blocker" if _n == 23
replace demoname = "P2Y12i" if _n == 24
replace demoname = "Statin" if _n == 25
replace demoname = "ACEi" if _n == 26
replace demoname = "ARB" if _n == 27
replace demoname = "ACEi or ARB" if _n == 28
replace demoname = "Beta blocker" if _n == 29
replace demoname = "No medication dispensed"  if _n == 30
replace demoname = "One medication class dispensed"  if _n == 31
replace demoname = "Two medication classes dispensed" if _n == 32
replace demoname = "Three medication classes dispensed"  if _n == 33
replace demoname = "Four medication classes dispensed" if _n == 34

drop HAtotal Atotal MAtotal
order demoname total totalproportion HA HAproportion A Aproportion MA MAproportion
save totpoparia, replace

}
{
*Create IRSD table data
use MI_ADS_ALL_age10, clear
su(IRSD_score), detail
matrix B = (r(p50)\r(p25)\r(p75))
su(IRSD_score) if STEMI == 0, detail
matrix B = (B,(r(p50)\r(p25)\r(p75)))
su(IRSD_score) if STEMI == 1, detail
matrix B = (B,(r(p50)\r(p25)\r(p75)))
su(IRSD_score) if ARIA == "HA", detail
matrix B = (B,(r(p50)\r(p25)\r(p75)))
su(IRSD_score) if ARIA == "A", detail
matrix B = (B,(r(p50)\r(p25)\r(p75)))
su(IRSD_score) if ARIA == "MA", detail
matrix B = (B,(r(p50)\r(p25)\r(p75)))
mat list B
clear
svmat B
foreach i in B1 B2 B3 B4 B5 B6 {
gen `i´s = string(`i´)
drop `i´
gen `i´ = `i´ + " (" + `i´[_n+1] + "-" + `i´[_n+2] + ")"
drop `i´s
}
drop if _n != 1
rename (B1 B2 B3 B4 B5 B6) (total NSTEMI STEMI HA A MA)
gen demoname = "Median IRSD score (IQR)"
order demoname
save irsdtot, replace
}
*Merge tables
{
use totalpopacs, clear
merge 1:1 demoname using totpoparia
sort id
drop id _merge
foreach i in total NSTEMI STEMI HA A MA {
gen `i´s = string(`i´)
drop `i´
rename `i´s `i´
}
foreach i in total NSTEMI STEMI HA A MA {
gen `i´s = `i´ + " " + "(" + `i´proportion + ")"
drop `i´ `i´proportion
rename `i´s `i´
}
*add in median IRSD data
append using irsdtot
gen id = _n
replace id = 16.5 if id == 35
```

```
sort id
drop id
gen category = ""
replace category = "Baseline characteristics" if _n == 2
replace category = "Co-morbidities" if _n == 10
replace category = "Revascularisation strategy" if _n == 15
replace category = "Socio-economic status" if _n == 17
replace category = "Medications dispensed within 90 days prior to MI admission" if _n == 20
replace category = "Medications dispensed within 90 days following MI admission" if _n == 25
order category
save totdemotable, replace
use totdemotable, clear
br
}
```

6.2.2 Tables of analysed population characteristics

This table includes the total analysed cohort, which excludes non-victorians and those who died during admission or within 90 days of discharge. There are two different tables, with the first showing characteristics by NSTEMI/STEMI diagnosis and remoteness category, and the second table by remoteness cateogory only, but with a breakdown of NSTEMI/STEMI within the categories.

```
use MI_ADS_ALL_age10, clear
br
gen tahelp = 1
*drop if died within 90 days
drop if hosp_mort != 0

ta tahelp STEMI, matcell(B1)
ta sex STEMI, matcell(B2)
ta agegroup STEMI, matcell(B3)
ta HT STEMI if HT==1, matcell(B5)
ta AF STEMI if AF==1, matcell (B6)
ta DM STEMI if DM==1, matcell(B7)
ta HF STEMI if HF==1, matcell (B8)
ta IS STEMI if IS==1, matcell (B9)
ta PCI_tag STEMI if PCI_tag==1, matcell (B10)
ta CABG_tag STEMI if CABG_tag==1, matcell (B11)
ta prior_P2Y12 STEMI if prior_P2Y12 == 1 , matcell (B12)
ta prior_statin STEMI if prior_statin == 1, matcell (B13)
ta prior_ace STEMI if prior_ace == 1, matcell (B14)
ta prior_arb STEMI if prior_arb == 1, matcell (B15)
ta prior_beta STEMI if prior_beta == 1, matcell (B16)
ta P2Y12_90d STEMI if P2Y12_90d==1 , matcell (B17)
ta statin_90d STEMI if statin_90d==1, matcell (B18)
ta ace_90d STEMI if ace_90d==1, matcell (B19)
ta arb_90d STEMI if arb_90d==1, matcell (B20)
ta ace_arb_90d STEMI if ace_arb_90d == 1, matcell(B23)
ta beta_90d STEMI if beta_90d==1, matcell (B21)
ta OMT_score STEMI, matcell(B22)

matrix analpop = (B1\B2\B3\B5\B6\B7\B8\B9\B10\B11\B12\B13\B14\B15\B16\B17\B18\B19\B20\B23\B21\B22)
matrix list analpop
clear
svmat analpop
br

rename (analpop1 analpop2) (NSTEMI STEMI)
egen total = max(NSTEMI + STEMI)
gen totalproportion = string((100 * (STEMI + NSTEMI) / total), "%3.0f")+"%"
replace total = NSTEMI + STEMI

foreach i in STEMI NSTEMI {
egen `i´total = max(`i´)
gen `i´proportion1 = (100 * `i´ / `i´total)
replace `i´proportion1 = `i´total / total * 100 if `i´proportion1 == 100
gen `i´proportion = string(`i´proportion1, "%3.0f")+"%"
drop `i´proportion1
```

```
}
gen id = _n
gen demoname = ""
replace demoname = "Male" if _n== 2
replace demoname = "Female" if _n== 3
replace demoname = "Aged 30-39" if _n== 4
replace demoname = "Aged 40-49" if _n== 5
replace demoname = "Aged 50-54" if _n== 6
replace demoname = "Aged 60-64" if _n== 7
replace demoname = "Aged 70-74" if _n== 8
replace demoname = "Aged 80+" if _n== 9
replace demoname = "Hypertension" if _n== 10
replace demoname = "Atrial fibrillation" if _n== 11
replace demoname = "Diabetes mellitus" if _n== 12
replace demoname = "Heart failure" if _n== 13
replace demoname = "Ischaemic stroke" if _n== 14
replace demoname = "PCI within 90 days of MI" if _n== 15
replace demoname = "CABG within 90 days of MI" if _n== 16
replace demoname = "prior P2Y12i" if _n == 17
replace demoname = "prior Statin" if _n == 18
replace demoname = "prior ACEi" if _n == 19
replace demoname = "prior ARB" if _n == 20
replace demoname = "prior Beta blocker" if _n == 21
replace demoname = "P2Y12i" if _n == 22
replace demoname = "Statin" if _n == 23
replace demoname = "ACEi" if _n == 24
replace demoname = "ARB" if _n == 25
replace demoname = "ACEi or ARB" if _n == 26
replace demoname = "Beta blocker" if _n == 27
replace demoname = "No medication dispensed"  if _n == 28
replace demoname = "One medication class dispensed"  if _n == 29
replace demoname = "Two medication classes dispensed" if _n == 30
replace demoname = "Three medication classes dispensed"  if _n == 31
replace demoname = "Four medication classes dispensed" if _n == 32

drop NSTEMItotal STEMItotal
order demoname total totalproportion NSTEMI NSTEMIproportion STEMI STEMIproportion
save analpopACS, replace
}
*Create table with ARIA
{
use MI_ADS_ALL_age10, clear
br
gen tahelp = 1
*drop if died within 90 days
drop if hosp_mort != 0

ta tahelp ARIA, matcell(B1)
ta sex ARIA, matcell(B2)
ta agegroup ARIA, matcell(B3)
ta HT ARIA if HT==1, matcell(B5)
ta AF ARIA if AF==1, matcell (B6)
ta DM ARIA if DM==1, matcell(B7)
ta HF ARIA if HF==1, matcell (B8)
ta IS ARIA if IS==1, matcell (B9)
ta PCI_tag ARIA if PCI_tag==1, matcell (B10)
ta CABG_tag ARIA if CABG_tag==1, matcell (B11)
ta prior_P2Y12 ARIA if prior_P2Y12 == 1 , matcell (B12)
ta prior_statin ARIA if prior_statin == 1, matcell (B13)
ta prior_ace ARIA if prior_ace == 1, matcell (B14)
ta prior_arb ARIA if prior_arb == 1, matcell (B15)
ta prior_beta ARIA if prior_beta == 1, matcell (B16)
ta P2Y12_90d ARIA if P2Y12_90d==1 , matcell (B17)
ta statin_90d ARIA if statin_90d==1, matcell (B18)
ta ace_90d ARIA if ace_90d==1, matcell (B19)
ta arb_90d ARIA if arb_90d==1, matcell (B20)
ta ace_arb_90d ARIA if ace_arb_90d == 1, matcell(B23)
ta beta_90d ARIA if beta_90d==1, matcell (B21)
ta OMT_score ARIA, matcell(B22)
```

```
matrix analpoparia = (B1\B2\B3\B5\B6\B7\B8\B9\B10\B11\B12\B13\B14\B15\B16\B17\B18\B19\B20\B23\B21\B2
> 2)
matrix list analpoparia
clear
svmat analpoparia
br

rename (analpoparia1 analpoparia2 analpoparia3) (A HA MA)
order HA A MA
egen total = max(HA+A+MA)
gen totalproportion = string((100 * (HA+A+MA) / total), "%3.0f")+"%"
replace total = HA+A+MA

foreach i in HA A MA {
egen `i´total = max(`i´)
gen `i´proportion1 = (100 * `i´ / `i´total)
replace `i´proportion1 = `i´total / total * 100 if `i´proportion1 == 100
gen `i´proportion = string(`i´proportion1, "%3.0f")+"%"
drop `i´proportion1
}

gen id = _n
gen demoname = ""
replace demoname = "Male" if _n== 2
replace demoname = "Female" if _n== 3
replace demoname = "Aged 30-39" if _n== 4
replace demoname = "Aged 40-49" if _n== 5
replace demoname = "Aged 50-54" if _n== 6
replace demoname = "Aged 60-64" if _n== 7
replace demoname = "Aged 70-74" if _n== 8
replace demoname = "Aged 80+" if _n== 9
replace demoname = "Hypertension" if _n== 10
replace demoname = "Atrial fibrillation" if _n== 11
replace demoname = "Diabetes mellitus" if _n== 12
replace demoname = "Heart failure" if _n== 13
replace demoname = "Ischaemic stroke" if _n== 14
replace demoname = "PCI within 90 days of MI" if _n== 15
replace demoname = "CABG within 90 days of MI" if _n== 16
replace demoname = "prior P2Y12i" if _n == 17
replace demoname = "prior Statin" if _n == 18
replace demoname = "prior ACEi" if _n == 19
replace demoname = "prior ARB" if _n == 20
replace demoname = "prior Beta blocker" if _n == 21
replace demoname = "P2Y12i" if _n == 22
replace demoname = "Statin" if _n == 23
replace demoname = "ACEi" if _n == 24
replace demoname = "ARB" if _n == 25
replace demoname = "ACEi or ARB" if _n == 26
replace demoname = "Beta blocker" if _n == 27
replace demoname = "No medication dispensed"  if _n == 28
replace demoname = "One medication class dispensed"  if _n == 29
replace demoname = "Two medication classes dispensed" if _n == 30
replace demoname = "Three medication classes dispensed"  if _n == 31
replace demoname = "Four medication classes dispensed" if _n == 32

drop HAtotal Atotal MAtotal
order demoname total totalproportion HA HAproportion A Aproportion MA MAproportion
save analpoparia, replace

*Create IRSD table data
use MI_ADS_ALL_age10, clear
drop if hosp_mort != 0
su(IRSD_score), detail
matrix B = (r(p50)\r(p25)\r(p75))
su(IRSD_score) if STEMI == 0, detail
matrix B = (B,(r(p50)\r(p25)\r(p75)))
su(IRSD_score) if STEMI == 1, detail
matrix B = (B,(r(p50)\r(p25)\r(p75)))
su(IRSD_score) if ARIA == "HA", detail
matrix B = (B,(r(p50)\r(p25)\r(p75)))
```

```
su(IRSD_score) if ARIA == "A", detail
matrix B = (B,(r(p50)\r(p25)\r(p75)))
su(IRSD_score) if ARIA == "MA", detail
matrix B = (B,(r(p50)\r(p25)\r(p75)))
mat list B
clear
svmat B
foreach i in B1 B2 B3 B4 B5 B6 {
gen `i´s = string(`i´)
drop `i´
gen `i´ = `i´ + " (" + `i´[_n+1] + "-" + `i´[_n+2] + ")"
drop `i´s
}
drop if _n != 1
rename (B1 B2 B3 B4 B5 B6) (total NSTEMI STEMI HA A MA)
gen demoname = "Median IRSD score (IQR)"
order demoname
save irsdanal, replace

*Merge tables
{
use analpopacs, clear
merge 1:1 demoname using analpoparia
sort id
drop id _merge
foreach i in total NSTEMI STEMI HA A MA {
gen `i´s = string(`i´)
drop `i´
rename `i´s `i´
}
foreach i in total NSTEMI STEMI HA A MA {
gen `i´s = `i´ + " " + "(" + `i´proportion + ")"
drop `i´ `i´proportion
rename `i´s `i´
}
*add in median IRSD
append using irsdtot
gen id = _n
replace id = 16.5 if id == 33
sort id
drop id
gen category = ""
replace category = "Baseline characteristics" if _n == 2
replace category = "Co-morbidities" if _n == 10
replace category = "Revascularisation strategy" if _n == 15
replace category = "Socio-economic status" if _n == 17
replace category = "Medications dispensed within 90 days prior to MI admission" if _n == 18
replace category = "Medications dispensed within 90 days post MI admission" if _n == 23
order category
save analdemotable, replace
use analdemotable, clear
br
}
*Table with ARIA only and STEMI breakdown
{
use MI_ADS_ALL_age10, clear
br
gen tahelp = 1
*drop if died within 90 days
drop if hosp_mort != 0

ta tahelp ARIA, matcell(B1)
ta STEMI ARIA if STEMI == 1, matcell(B23)
ta STEMI ARIA if STEMI == 0, matcell(B24)
ta sex ARIA, matcell(B2)
ta agegroup ARIA, matcell(B3)
ta HT ARIA if HT==1, matcell(B5)
ta AF ARIA if AF==1, matcell (B6)
ta DM ARIA if DM==1, matcell(B7)
```

```
ta HF ARIA if HF==1, matcell (B8)
ta IS ARIA if IS==1, matcell (B9)
ta PCI_tag ARIA if PCI_tag==1, matcell (B10)
ta CABG_tag ARIA if CABG_tag==1, matcell (B11)
ta prior_P2Y12 ARIA if prior_P2Y12 == 1 , matcell (B12)
ta prior_statin ARIA if prior_statin == 1, matcell (B13)
ta prior_ace ARIA if prior_ace == 1, matcell (B14)
ta prior_arb ARIA if prior_arb == 1, matcell (B15)
ta prior_ace_arb ARIA if prior_ace_arb == 1, matcell(B25)
ta prior_beta ARIA if prior_beta == 1, matcell (B16)
ta P2Y12_90d ARIA if P2Y12_90d==1 , matcell (B17)
ta statin_90d ARIA if statin_90d==1, matcell (B18)
ta ace_90d ARIA if ace_90d==1, matcell (B19)
ta arb_90d ARIA if arb_90d==1, matcell (B20)
ta ace_arb_90d ARIA if ace_arb_90d == 1, matcell(B26)
ta beta_90d ARIA if beta_90d==1, matcell (B21)
ta OMT_score ARIA, matcell(B22)
ta P2Y1280adj ARIA if P2Y1280adj == 1, matcell(B27)
ta statin80adj ARIA if statin80adj == 1, matcell(B28)
ta ACE_ARB80adj ARIA if ACE_ARB80adj == 1, matcell(B29)
ta beta80adj ARIA if beta80adj == 1, matcell(B30)

matrix analpoparia = (B1\B23\B24\B2\B3\B5\B6\B7\B8\B9\B10\B11\B12\B13\B14\B15\B25\B16\B17\B18\B19\B2
> 0\B26\B21\B22\B27\B28\B29\B30)
matrix list analpoparia
clear
svmat analpoparia
br

rename (analpoparia1 analpoparia2 analpoparia3) (A HA MA)
order HA A MA
egen total = max(HA+A+MA)
gen totalproportion = string((100 * (HA+A+MA) / total), "%3.0f")+"%"
replace total = HA+A+MA

foreach i in HA A MA {
egen `i´total = max(`i´)
gen `i´proportion1 = (100 * `i´ / `i´total)
replace `i´proportion1 = `i´total / total * 100 if `i´proportion1 == 100
gen `i´proportion = string(`i´proportion1, "%3.0f")+"%"
drop `i´proportion1
}

gen id = _n
gen demoname = ""
replace demoname = "STEMI"  if _n == 2
replace demoname = "NSTEMI" if _n == 3
replace demoname = "Male" if _n == 4
replace demoname = "Female" if _n== 5
replace demoname = "Aged 30-39" if _n== 6
replace demoname = "Aged 40-49" if _n== 7
replace demoname = "Aged 50-59" if _n== 8
replace demoname = "Aged 60-69" if _n== 9
replace demoname = "Aged 70-79" if _n==10
replace demoname = "Aged 80+" if _n== 11
replace demoname = "Hypertension" if _n == 12
replace demoname = "Atrial fibrillation" if _n== 13
replace demoname = "Diabetes mellitus" if _n== 14
replace demoname = "Heart failure" if _n== 15
replace demoname = "Ischaemic stroke" if _n== 16
replace demoname = "PCI within 90 days of MI" if _n== 17
replace demoname = "CABG within 90 days of MI" if _n== 18
replace demoname = "prior P2Y12i" if _n == 19
replace demoname = "prior Statin" if _n == 20
replace demoname = "prior ACEi" if _n == 21
replace demoname = "prior ARB" if _n == 22
replace demoname = "prior ACEi or ARB" if _n == 23
replace demoname = "prior Beta blocker" if _n == 24
replace demoname = "P2Y12i" if _n == 25
replace demoname = "Statin" if _n == 26
```

```
replace demoname = "ACEi" if _n == 27
replace demoname = "ARB" if _n == 28
replace demoname = "ACEi or ARB" if _n == 29
replace demoname = "Beta blocker" if _n == 30
replace demoname = "No medication dispensed"  if _n == 31
replace demoname = "One medication class dispensed"  if _n == 32
replace demoname = "Two medication classes dispensed" if _n == 33
replace demoname = "Three medication classes dispensed"  if _n == 34
replace demoname = "Four medication classes dispensed" if _n == 35
replace demoname = ">80% PDC P2Y12i" if _n == 36
replace demoname = ">80% PDC Statin" if _n == 37
replace demoname = ">80% PDC ACEi or ARB" if _n == 38
replace demoname = ">80% PDC Beta blocker" if _n == 39

drop HAtotal Atotal MAtotal id
order demoname total totalproportion HA HAproportion A Aproportion MA MAproportion
foreach i in total HA A MA {
gen `i´s = string(`i´)
drop `i´
rename `i´s `i´
}
foreach i in total HA A MA {
gen `i´s = `i´ + " " + "(" + `i´proportion + ")"
drop `i´ `i´proportion
rename `i´s `i´
}
order demoname total HA A MA
save analpopariastemi, replace
*Create IRSD table data
use MI_ADS_ALL_age10, clear
drop if hosp_mort != 0
su(IRSD_score), detail
matrix B = (r(p50)\r(p25)\r(p75))
su(IRSD_score) if ARIA == "HA", detail
matrix B = (B,(r(p50)\r(p25)\r(p75)))
su(IRSD_score) if ARIA == "A", detail
matrix B = (B,(r(p50)\r(p25)\r(p75)))
su(IRSD_score) if ARIA == "MA", detail
matrix B = (B,(r(p50)\r(p25)\r(p75)))
mat list B
clear
svmat B
foreach i in B1 B2 B3 B4 {
gen `i´s = string(`i´)
drop `i´
gen `i´ = `i´ + " (" + `i´[_n+1] + "-" + `i´[_n+2] + ")"
drop `i´s
}
drop if _n != 1
rename (B1 B2 B3 B4 ) (total HA A MA)
gen demoname = "Median IRSD score (IQR)"
order demoname
save irsdanal1, replace
}
*append IRSD table data
use analpopariastemi, clear
append using irsdanal1
gen id = _n
replace id = 18.5 if id == 40
sort id
drop id
gen category = ""
replace category = "Myocardial infarction (MI) diagnosis" if _n == 2
replace category = "Baseline characteristics" if _n == 4
replace category = "Co-morbidities" if _n == 12
replace category = "Revascularisation strategy" if _n == 17
replace category = "Socio-economic status" if _n == 19
replace category = "Medication dispensed within 90 days prior to MI" if _n == 20
replace category = "Medication dispensed within 90 days post MI" if _n == 25
```

```
replace category = "Medication adherence 12 months post MI" if _n == 37
order category
save analpopariastemi_irsd, replace
br
}
```

6.3 Data analysis

6.3.1 Regression analysis for dispensing using means for NSTEMI and STEMI cohorts for prediction (splines and full adjusted model)

We first modified age categoires to enable us to create splines of age, then generated mean values of sex, IRSD, HT, AF, HF, DM, CABG and PCI. These were selected from eac of the separate NSTEMI and STEMI stratified cohorts. Following this step, we created a dataset where these values where present for each ARIA value, including splines of ARIA values and age.
We selected a logistic regression model for this analysis, using splines of ARIA as a continuous variable to account for non-linearity. The model was adjusted for multiple covariates; with and spline effects of age; categorical variables of: sex, the presence of hypertension, atrial fibrillation, diabetes, heart failure, and ischaemic stroke, as well as and revascularisation strategy of PCI or CABG within 30 days post MI; and socio-economic status using IRSD score as continuous variable. Using the model, we predcited probabilities of dispensing for NSTEMI with the following mean values were dervided from the NSTEMI stratified cohort:

- Mean age 70.93
- Sex 1.36 [male = 1 female = 2]
- IRSD score 997.87
- Presence of hypertension 0.65 (binary)
- Presence of AF 0.14 (binary)
- Presence of heart failure 0.17 (binary)
- Presence of diabetes 0.32 (binary)
- Presence of ischaemic stroke 0.11 (binary)
- received PCI within 30 days of admission 0.33 (binary)
- Received CABG within 30 days of admission (0.11)

Using the model above, we predcited probabilities of dispensing for STEMI with the following mean values derived from the STEMI stratified cohort:

- Mean age 64.24
- Sex 1.25 [male = 1 female = 2]
- IRSD score 999.77

- Presence of hypertension 0.54 (binary)
- Presence of AF 0.12 (binary)
- Presence of heart failure 0.14 (binary)
- Presence of diabetes 0.22 (binary)
- Presence of ischaemic stroke 0.08 (binary)
- received PCI within 30 days of admission 0.75 (binary)
- Received CABG within 30 days of admission 0.08 (binary)

```
*Create age group sections to allow for age spline creation
{
use MI_ADS_ALL, clear
drop if hosp_mort != 0
br
gen agespline = agegroup + 2.5 if agegroup != 85
replace agespline = agegroup + 5 if agegroup == 85
save MI_ADS_ALL_spline, replace
}
*Creation of prediction data set, using marginal effects at the means of independent variables for N
> STEMI and STEMI.
{
forval ii = 0/1 {
use MI_ADS_ALL_spline, clear
foreach i in agespline sex IRSD_score HT AF HF DM IS CABG_tag PCI_tag {
su(`i´) if STEMI == `ii´
local m`i´ = r(mean)
}
clear
set obs 49
gen mean_ARIA = (_n-1)/10
br
foreach i in agespline sex IRSD_score  HT AF HF DM IS CABG_tag PCI_tag {
gen `i´ = `m`i´´
}
mkspline ARIAS= mean_ARIA, cubic knots(0 0.05 0.5 2)
mkspline ages = agespline, cubic knots(32.5 62.5 72.5 77.5)
save analysis_adherence_set_`ii´, replace
}
}
*Set up splines followed by logistic regression model construction for NSTEMI and STEMI groups
{
forval ii = 0/1 {
foreach i in beta_90d ace_arb_90d statin_90d P2Y12_90d {
use MI_ADS_ALL_spline, clear
*create splines for remoteness
mkspline ARIAS = mean_ARIA, cubic knots(0 0.05 0.5 2)
*create splines for ages
mkspline ages = agegroup, cubic knots(32.5 62.5 72.5 77.5)
drop agespline
logistic `i´ c.ARIAS* c.ages* sex IRSD_score HT AF HF DM IS CABG_tag PCI_tag if STEMI ==`ii´, coef
use analysis_adherence_set_`ii´, clear
*create prediction variable from model, then transform into probabability
predict A
predict xb, xb
*create SE to build confidence intervals
predict B, stdp
gen ll = xb - (invnormal(0.975)*B)
gen ul = xb + (invnormal(0.975)*B)
```

```
replace ll = invlogit(ll)
replace ul = invlogit(ul)
gen STEMI = `ii´
keep mean_ARIA A ul ll STEMI
save `i´_`ii´, replace
}
}
}
*Create figure of predicted probability of dispensing according ARIA
{
*NSTEMI
foreach i in beta_90d ace_arb_90d statin_90d P2Y12_90d {
use `i´_0, clear
twoway ///
(rarea ul ll mean_ARIA if STEMI == 0, col(navy%30) fintensity(inten80) lwidth(none)) ///
(line A mean_ARIA if STEMI == 0, col(navy)) ///
, graphregion(color(white)) ///
xtitle("mean ARIA score") ytitle("Predicted probability of dispensing at 90 days post discharge") //
> /
legend(order(2 "NSTEMI") position(6) ring(0) row(1) col(2) region(lcolor(white) color(none))) ///
yscale(range(0.5 1)) ylabel(0.5 "0.5" 0.6 "0.6" 0.7 "0.7" 0.8 "0.8" 0.9 "0.9" 1.0 "1.0" , angle(0) f
> ormat(%9.0f)) xscale(range(0 5)) ///
title("`i´", placement(west) color(black) size(medium))
graph save "Graph" fig_receipt_`i´_0, replace
}
*STEMI
foreach i in beta_90d ace_arb_90d statin_90d P2Y12_90d {
use `i´_1, clear
twoway ///
(rarea ul ll mean_ARIA if STEMI == 1, col(red%30) fintensity(inten80) lwidth(none)) ///
(line A mean_ARIA if STEMI == 1 , col(red)) ///
, graphregion(color(white)) ///
xtitle("mean ARIA score") ytitle("Predicted probability of dispensing at 90 days post discharge") //
> /
legend(order(2 "STEMI") position(6) ring(0) row(1) col(2) region(lcolor(white) color(none))) ///
yscale(range(0.5 1)) ylabel(0.5 "0.5" 0.6 "0.6" 0.7 "0.7" 0.8 "0.8" 0.9 "0.9" 1.0 "1.0" , angle(0) f
> ormat(%9.0f)) xscale(range(0 5)) ///
title("`i´", placement(west) color(black) size(medium))
graph save "Graph" fig_receipt_`i´_1, replace
}
}
*Combine all graphs
{
graph combine ///
fig_receipt_P2Y12_90d_1.gph ///
fig_receipt_P2Y12_90d_0.gph ///
fig_receipt_statin_90d_1.gph ///
fig_receipt_statin_90d_0.gph ///
fig_receipt_beta_90d_1.gph ///
fig_receipt_beta_90d_0.gph ///
fig_receipt_ace_arb_90d_1.gph ///
fig_receipt_ace_arb_90d_0.gph ///
, altshrink cols(2) graphregion(color(white)) xsize(4.5)
graph export "G:\Adam\Project 2  - location and MI outcomes\Results\Fig_dispensing_STEMI_NSTEMI.pdf"
> , as(pdf) name("Graph") replace
}
}
```

6.3.2 Regression analysis for dispensing using means for total cohort for prediction (splines and full adjusted model)

This step replicated the method above, however we took the means of of sex, IRSD, HT, AF, HF, DM, CABG and PCI for the total analysed cohort and applied it STEMI and NSTEMI respectively. This is presented in the supplement of the manuscript as a secondary analysis. Using the model,

we predcited probabilities of dispensing for NSTEMI and STEMI with the following mean values were derived from the total analysed cohort:

- Mean age 69.20
- Sex 1.33 [male = 1 female = 2]
- IRSD score 998.36
- Presence of hypertension 0.62 (binary)
- Presence of AF 0.14 (binary)
- Presence of heart failure 0.16 (binary)
- Presence of diabetes 0.30 (binary)
- Presence of ischaemic stroke 0.10 (binary)
- received PCI within 30 days of admission 0.44 (binary)
- Received CABG within 30 days of admission (0.10)

```
*Creation of prediction data set, using marginal effects at the means of independent variables for N
> STEMI and STEMI.
{
use MI_ADS_ALL_spline, clear
foreach i in agespline sex IRSD_score HT AF HF DM IS CABG_tag PCI_tag {
su(`i´)
local m`i´ = r(mean)
}
clear
set obs 49
gen mean_ARIA = (_n-1)/10
br
foreach i in agespline sex IRSD_score  HT AF HF DM IS CABG_tag PCI_tag {
gen `i´ = `m`i´´
}
mkspline ARIAS= mean_ARIA, cubic knots(0 0.05 0.5 2)
mkspline ages = agespline, cubic knots(32.5 62.5 72.5 77.5)
save analysis_adherence_set_total, replace
}
}
*Set up splines followed by logistic regression model construction for NSTEMI and STEMI groups
{
forval ii = 0/1 {
foreach i in beta_90d ace_arb_90d statin_90d P2Y12_90d {
use MI_ADS_ALL_spline, clear
*create splines for remoteness
mkspline ARIAS = mean_ARIA, cubic knots(0 0.05 0.5 2)
*create splines for ages
mkspline ages = agegroup, cubic knots(32.5 62.5 72.5 77.5)
drop agespline
logistic `i´ c.ARIAS* c.ages* sex IRSD_score HT AF HF DM IS CABG_tag PCI_tag if STEMI ==`ii´, coef

use analysis_adherence_set_total, clear
*create prediction variable from model, then transform into probabability
predict A
predict xb, xb
*create SE to build confidence intervals
```

```
predict B, stdp
gen ll = xb - (invnormal(0.975)*B)
gen ul = xb + (invnormal(0.975)*B)
replace ll = invlogit(ll)
replace ul = invlogit(ul)
gen STEMI = `ii´
keep mean_ARIA A ul ll STEMI
save `i´_`ii´_total, replace
}
}
}
*Create figure of predicted probability of dispensing according ARIA
{
foreach i in beta_90d ace_arb_90d statin_90d P2Y12_90d {
use `i´_0_total, clear
append using `i´_1_total
twoway ///
(rarea ul ll mean_ARIA if STEMI == 0, col(navy%30) fintensity(inten80) lwidth(none)) ///
(line A mean_ARIA if STEMI == 0, col(navy)) ///
(rarea ul ll mean_ARIA if STEMI == 1, col(red%30) fintensity(inten80) lwidth(none)) ///
(line A mean_ARIA if STEMI == 1 , col(red)) ///
, graphregion(color(white)) ///
xtitle("mean ARIA score") ytitle("Predicted probability of dispensing at 90 days post discharge") //
> /
legend(order(2 "NSTEMI") position(6) ring(0) row(1) col(2) region(lcolor(white) color(none))) ///
yscale(range(0.5 1)) ylabel(0.5 "0.5" 0.6 "0.6" 0.7 "0.7" 0.8 "0.8" 0.9 "0.9" 1.0 "1.0" , angle(0) f
> ormat(%9.0f)) xscale(range(0 5)) ///
title("`i´", placement(west) color(black) size(medium))
graph save "Graph" fig_receipt_`i´_total, replace
}
}
*Combine all graphs
{
graph combine ///
fig_receipt_P2Y12_90d_total.gph ///
fig_receipt_statin_90d_total.gph ///
fig_receipt_beta_90d_total.gph ///
fig_receipt_ace_arb_90d_total.gph ///
, altshrink cols(2) graphregion(color(white)) xsize(4.5)
graph export "G:\Adam\Project 2  - location and MI outcomes\Results\Fig_dispensing_total.pdf", as(pd
> f) name("Graph") replace
}
}
```

6.3.3 Regression analysis for PDC using means for NSTEMI and STIME cohorts for prediction (splines and fully adjusted model only)

We used the same method for dispensing in terms of splines of ARIA and age, as well as mean values of of sex, IRSD, HT, AF, HF, DM, CABG and PCI. We ran the analysis twice like in the dispensing analysis, one with mean values bvased on NSTEMI and STEMI straitifed cohorts, and the other using mean values from the total analysed cohort.
We selected a fractional logistic regression model with a logit link function for this analysis, using splines of ARIA as a continuous variable to account for non-linearity. The model was adjusted for multiple covariates; with and spline effects of age; categorical variables of: sex, the presence of hypertension, atrial fibrillation, diabetes, heart failure, and ischaemic stroke, as well as and revascularisation strategy of PCI or CABG within 30 days post MI; and socio-economic status using IRSD score as continuous variable.

Using the model above, we predcited PDC for NSTEMI with the following mean values were dervided from the NSTEMI stratified cohort:

NSTEMI

- Mean age 70.93
- Sex 1.36 [male = 1 female = 2]
- IRSD score 997.87
- Presence of hypertension 0.65 (binary)
- Presence of AF 0.14 (binary)
- Presence of heart failure 0.17 (binary)
- Presence of diabetes 0.32 (binary)
- Presence of ischaemic stroke 0.11 (binary)
- received PCI within 30 days of admission 0.33 (binary)
- Received CABG within 30 days of admission (0.11)

Using the model above, we predcited PDC for STEMI with the following mean values derived from the STEMI stratified cohort:

STEMI

- Mean age 64.24
- Sex 1.25 [male = 1 female = 2]
- IRSD score 999.77
- Presence of hypertension 0.54 (binary)
- Presence of AF 0.12 (binary)
- Presence of heart failure 0.14 (binary)
- Presence of diabetes 0.22 (binary)
- Presence of ischaemic stroke 0.08 (binary)
- received PCI within 30 days of admission 0.75 (binary)
- Received CABG within 30 days of admission 0.08 (binary)

```
*We will use the analysis adherence set created containing spline values and means of co-variates fo
> r the predict command below.
*Set up splines and offest followed by fractional regression model construction logit function.
forval ii = 0/1 {
foreach i in PDC_adj_beta PDC_adj_ACE_ARB PDC_adj_statins PDC_adj_P2Y12 {
use MI_ADS_ALL_spline, clear
*create splines for remoteness
mkspline ARIAS = mean_ARIA, cubic knots(0 0.05 0.5 2)
*create splines for ages
mkspline ages = agegroup, cubic knots(32.5 62.5 72.5 77.5)
drop agespline
fracreg logit `i´ c.ARIAS* c.ages* sex IRSD_score HT AF HF DM IS CABG_tag PCI_tag if STEMI ==`ii´
}

use analysis_adherence_set_`ii´, clear
*create prediction variable from model, then transform into predicted PDC
predict A
predict xb, xb
*create SE to build confidence intervals
predict B, stdp
gen ll = xb - (invnormal(0.975)*B)
gen ul = xb + (invnormal(0.975)*B)
replace ll = invlogit(ll)
replace ul = invlogit(ul)
gen STEMI = `ii´
keep mean_ARIA A ul ll STEMI
save `i´_`ii´_adherence, replace
}
}
*Create figure
{
*NSTEMI
foreach i in PDC_adj_beta PDC_adj_ACE_ARB PDC_adj_statins PDC_adj_P2Y12 {
use `i´_0_adherence, clear
twoway ///
(rarea ul ll mean_ARIA if STEMI == 0, col(navy%30) fintensity(inten80) lwidth(none)) ///
(line A mean_ARIA if STEMI == 0, col(navy)) ///
, graphregion(color(white)) ///
xtitle("mean ARIA score") ytitle("Predicted mean PDC at 12 months") ///
legend(order(4 "NSTEMI") position(6) ring(0) row(1) col(2) region(lcolor(white) color(none))) ///
yscale(range(0.5 1)) ylabel(0.5 "0.5" 0.6 "0.6" 0.7 "0.7" 0.8 "0.8" 0.9 "0.9" 1.0 "1.0" , angle(0) f
> ormat(%9.0f)) xscale(range(0 5)) ///
title("`i´", placement(west) color(black) size(medium))
graph save "Graph" fig_`i´_0, replace
}
*STEMI
foreach i in PDC_adj_beta PDC_adj_ACE_ARB PDC_adj_statins PDC_adj_P2Y12 {
use `i´_1_adherence, clear
twoway ///
(rarea ul ll mean_ARIA if STEMI == 1, col(red%30) fintensity(inten80) lwidth(none)) ///
(line A mean_ARIA if STEMI == 1 , col(red)) ///
, graphregion(color(white)) ///
xtitle("mean ARIA score") ytitle("Predicted mean PDC at 12 months") ///
legend(order(4 "STEMI") position(6) ring(0) row(1) col(2) region(lcolor(white) color(none))) ///
yscale(range(0.5 1)) ylabel(0.5 "0.5" 0.6 "0.6" 0.7 "0.7" 0.8 "0.8" 0.9 "0.9" 1.0 "1.0" , angle(0) f
> ormat(%9.0f)) xscale(range(0 5)) ///
title("`i´", placement(west) color(black) size(medium))
graph save "Graph" fig_`i´_1, replace
}
*Combine all graphs
{
graph combine ///
fig_PDC_adj_P2Y12_1.gph ///
fig_PDC_adj_P2Y12_0.gph ///
fig_PDC_adj_statins_1.gph ///
fig_PDC_adj_statins_0.gph ///
fig_PDC_adj_beta_1.gph ///
```

```
fig_PDC_adj_beta_0.gph ///
fig_PDC_adj_ACE_ARB_1.gph ///
fig_PDC_adj_ACE_ARB_0.gph ///
, altshrink cols(2) graphregion(color(white)) xsize(4.5)
graph export "G:\Adam\Project 2  - location and MI outcomes\Results\Fig_pdc_NSTEMI_STEMI.pdf", as(pd
> f) name("Graph") replace
}
}
```

6.3.4 Regression analysis for PDC using means for total analysed population for prediction (splines and fully adjusted model only)

This analysis was the same as above, but like with the dispensing analysis, we predicted at the mean values of sex, IRSD, HT, AF, HF, DM, CABG and PCI for the total analysed cohort. This is presented in the supplement of the manuscript as a secondary analysis.
Using the model, we predcited probabilities of dispensing for NSTEMI and STEMI with the following mean values were derived from the total analysed cohort:

- Mean age 69.20
- Sex 1.33 [male = 1 female = 2]
- IRSD score 998.36
- Presence of hypertension 0.62 (binary)
- Presence of AF 0.14 (binary)
- Presence of heart failure 0.16 (binary)
- Presence of diabetes 0.30 (binary)
- Presence of ischaemic stroke 0.10 (binary)
- received PCI within 30 days of admission 0.44 (binary)
- Received CABG within 30 days of admission (0.10)

```
*We will use the analysis adherence set created containing spline values and means of co-variates fo
> r the predict command below.
*Set up splines and offest followed by fractional regression model construction logit function.
forval ii = 0/1 {
foreach i in PDC_adj_beta PDC_adj_ACE_ARB PDC_adj_statins PDC_adj_P2Y12 {
use MI_ADS_ALL_spline, clear
*create splines for remoteness
mkspline ARIAS = mean_ARIA, cubic knots(0 0.05 0.5 2)
*create splines for ages
mkspline ages = agegroup, cubic knots(32.5 62.5 72.5 77.5)
drop agespline
fracreg logit `i´ c.ARIAS* c.ages* sex IRSD_score HT AF HF DM IS CABG_tag PCI_tag if STEMI ==`ii´
use analysis_adherence_set_total, clear
*create prediction variable from model, then transform into probabability
predict A
predict xb, xb
*create SE to build confidence intervals
predict B, stdp
gen ll = xb - (invnormal(0.975)*B)
```

```
gen ul = xb + (invnormal(0.975)*B)
replace ll = invlogit(ll)
replace ul = invlogit(ul)
gen STEMI = `ii´
keep mean_ARIA A ul ll STEMI
save `i´_`ii´_adherence_total, replace
}
}
*Create figure per drug class
{
foreach i in PDC_adj_beta PDC_adj_ACE_ARB PDC_adj_statins PDC_adj_P2Y12 {
use `i´_0_adherence_total, clear
append using `i´_1_adherence_total
twoway ///
(rarea ul ll mean_ARIA if STEMI == 1, col(red%30) fintensity(inten80) lwidth(none)) ///
(line A mean_ARIA if STEMI == 1 , col(red)) ///
(rarea ul ll mean_ARIA if STEMI == 0, col(navy%30) fintensity(inten80) lwidth(none)) ///
(line A mean_ARIA if STEMI == 0, col(navy)) ///
, graphregion(color(white)) ///
xtitle("mean ARIA score") ytitle("Predicted mean PDC at 12 months") ///
legend(order(4 "STEMI" 2 "NSTEMI") position(6) ring(0) row(1) col(2) region(lcolor(white) color(none
> ))) ///
yscale(range(0 1)) ylabel(0 "0" 0.1 "0.1" 0.2 "0.2" 0.3 "0.3" 0.4 "0.4" 0.5 "0.5" 0.6 "0.6" 0.7 "0.7
> " 0.8 "0.8" 0.9 "0.9" 1.0 "1.0" , angle(0) format(%9.0f)) xscale(range(0 5)) ///
title("`i´", placement(west) color(black) size(medium))
graph save "Graph" fig_`i´_total, replace
}
*Combine all graphs
{
graph combine ///
fig_PDC_adj_P2Y12_total.gph ///
fig_PDC_adj_statins_total.gph ///
fig_PDC_adj_beta_total.gph ///
fig_PDC_adj_ACE_ARB_total.gph ///
, altshrink cols(2) graphregion(color(white)) xsize(4.5)
graph export "G:\Adam\Project 2  - location and MI outcomes\Results\Fig_pdc_total.pdf", as(pdf) name
> ("Graph") replace
}
}
}
```

6.4 Supplemental analysis

In the previous analysis, all MI admissions were considered, with those with no initial dispensing of medications included in PDC, with non-dispensing of therapy considered as non-adherent. This was based on the premise that all people with MI should eb initiated on secondary prevention therapy prior to discharge, and be continued for at least one year. This analysis instead excludes those where there was no initial disensing (within 90 days from discharge) occurred for each specific class of secondary prevention medication.
Predictions for proportion of days covered where still made at the means for the total stratified NSTEMI and STEMI cohorts:

NSTEMI

- Mean age 70.93
- Sex 1.36 [male = 1 female = 2]
- IRSD score 997.87

- Presence of hypertension 0.65 (binary)
- Presence of AF 0.14 (binary)
- Presence of heart failure 0.17 (binary)
- Presence of diabetes 0.32 (binary)
- Presence of ischaemic stroke 0.11 (binary)
- received PCI within 30 days of admission 0.33 (binary)
- Received CABG within 30 days of admission (0.11)

Using the model above, we predcited PDC for STEMI with the following mean values derived from the STEMI stratified cohort:

STEMI

- Mean age 64.24
- Sex 1.25 [male = 1 female = 2]
- IRSD score 999.77
- Presence of hypertension 0.54 (binary)
- Presence of AF 0.12 (binary)
- Presence of heart failure 0.14 (binary)
- Presence of diabetes 0.22 (binary)
- Presence of ischaemic stroke 0.08 (binary)
- received PCI within 30 days of admission 0.75 (binary)
- Received CABG within 30 days of admission 0.08 (binary)

We first need to create the spline data set for cohort where non-dispensing at 90 days is excluded from PDC analysis, then replicate the above analysis for PDC using NSTEMI and STEMI stratified cohorts.

```
use MI_ADS_ALL_SA, clear
drop if hosp_mort != 0
br
gen agespline = agegroup + 2.5 if agegroup != 85
replace agespline = agegroup + 5 if agegroup == 85
*We will use the analysis adherence set created containing spline values and means of co-variates fo
> r the predict command below.
*Set up splines and offest followed by fractional regression model construction logit function.
forval ii = 0/1 {
foreach i in PDC_adj_beta PDC_adj_ACE_ARB PDC_adj_statins PDC_adj_P2Y12 {
use MI_ADS_ALL_spline, clear
*create splines for remoteness
mkspline ARIAS = mean_ARIA, cubic knots(0 0.05 0.5 2)
*create splines for ages
mkspline ages = agegroup, cubic knots(32.5 62.5 72.5 77.5)
```

```
drop agespline
fracreg logit `i´ c.ARIAS* c.ages* sex IRSD_score HT AF HF DM IS CABG_tag PCI_tag if STEMI ==`ii´
}
use SAanalysis_adherence_set_`ii´, clear
*create prediction variable from model, then transform into predicted PDC
predict A
predict xb, xb
*create SE to build confidence intervals
predict B, stdp
gen ll = xb - (invnormal(0.975)*B)
gen ul = xb + (invnormal(0.975)*B)
replace ll = invlogit(ll)
replace ul = invlogit(ul)
gen STEMI = `ii´
keep mean_ARIA A ul ll STEMI
save SA`i´_`ii´_adherence, replace
}
}
*Create figure per drug class
{
*NSTEMI
foreach i in PDC_adj_beta PDC_adj_ACE_ARB PDC_adj_statins PDC_adj_P2Y12 {
use SA`i´_0_adherence, clear
twoway ///
(rarea ul ll mean_ARIA if STEMI == 0, col(navy%30) fintensity(inten80) lwidth(none)) ///
(line A mean_ARIA if STEMI == 0, col(navy)) ///
, graphregion(color(white)) ///
xtitle("mean ARIA score") ytitle("Predicted mean PDC at 12 months") ///
legend(order(4 "NSTEMI") position(6) ring(0) row(1) col(2) region(lcolor(white) color(none))) ///
yscale(range(0.5 1)) ylabel(0.5 "0.5" 0.6 "0.6" 0.7 "0.7" 0.8 "0.8" 0.9 "0.9" 1.0 "1.0" , angle(0) f
> ormat(%9.0f)) xscale(range(0 5)) ///
title("`i´", placement(west) color(black) size(medium))
graph save "Graph" fig_`i´_0, replace
}
*STEMI
foreach i in PDC_adj_beta PDC_adj_ACE_ARB PDC_adj_statins PDC_adj_P2Y12 {
use SA`i´_1_adherence, clear
twoway ///
(rarea ul ll mean_ARIA if STEMI == 1, col(red%30) fintensity(inten80) lwidth(none)) ///
(line A mean_ARIA if STEMI == 1 , col(red)) ///
, graphregion(color(white)) ///
xtitle("mean ARIA score") ytitle("Predicted mean PDC at 12 months") ///
legend(order(4 "STEMI") position(6) ring(0) row(1) col(2) region(lcolor(white) color(none))) ///
yscale(range(0.5 1)) ylabel(0.5 "0.5" 0.6 "0.6" 0.7 "0.7" 0.8 "0.8" 0.9 "0.9" 1.0 "1.0" , angle(0) f
> ormat(%9.0f)) xscale(range(0 5)) ///
title("`i´", placement(west) color(black) size(medium))
graph save "Graph" fig_`i´_1, replace
}
*Combine all graphs
{
graph combine ///
fig_PDC_adj_P2Y12_1.gph ///
fig_PDC_adj_P2Y12_0.gph ///
fig_PDC_adj_statins_1.gph ///
fig_PDC_adj_statins_0.gph ///
fig_PDC_adj_beta_1.gph ///
fig_PDC_adj_beta_0.gph ///
fig_PDC_adj_ACE_ARB_1.gph ///
fig_PDC_adj_ACE_ARB_0.gph ///
, altshrink cols(2) graphregion(color(white)) xsize(4.5)
graph export "G:\Adam\Project 2  - location and MI outcomes\Results\Fig_pdc_NSTEMI_STEMI_SA.pdf", as
> (pdf) name("Graph") replace
}
}
```

We then plotted both predicted PDC values for the total analysed cohort as well as the cohort who were dispensed at least one prescription in the first 90 days for each medication class.

```
*NSTEMI
foreach i in PDC_adj_beta PDC_adj_ACE_ARB PDC_adj_statins PDC_adj_P2Y12 {
use SA`i´_0_adherence, clear
gen SA = 1
append using `i´_0_adherence_SA, clear
replace SA =0 if SA ==.
twoway ///
(rarea ul ll mean_ARIA if SA == 1, col(navy%30) fintensity(inten80) lwidth(none)) ///
(line A mean_ARIA if sa == 1, col(navy)) ///
(rarea ul ll mean_ARIA if SA == 0, col(purple%30) fintensity(inten80) lwidth(none)) ///
(line A mean_ARIA if SA == 0, col(purple)) ///
, graphregion(color(white)) ///
xtitle("mean ARIA score") ytitle("Predicted mean PDC at 12 months") ///
legend(order(4 "NSTEMI- All MI admissions" 2 "NSTEMI-Excluding peoiple with no dispensing") position
> (6) ring(0) row(1) col(2) region(lcolor(white) color(none))) ///
yscale(range(0 1)) ylabel(0 "0" 0.1 "0.1" 0.2 "0.2" 0.3 "0.3" 0.4 "0.4" 0.5 "0.5" 0.6 "0.6" 0.7 "0.7
> " 0.8 "0.8" 0.9 "0.9" 1.0 "1.0" , angle(0) format(%9.0f)) xscale(range(0 5)) ///
title("`i´", placement(west) color(black) size(medium))
graph save "Graph" fig_`i´_0_SAcompare, replace
}
*STEMI
foreach i in PDC_adj_beta PDC_adj_ACE_ARB PDC_adj_statins PDC_adj_P2Y12 {
use SA`i´_1_adherence, clear
gen SA = 1
append using `i´_1_adherence_SA, clear
replace SA =0 if SA ==.
twoway ///
(rarea ul ll mean_ARIA if SA == 1, col(red%30) fintensity(inten80) lwidth(none)) ///
(line A mean_ARIA if sa == 1, col(red)) ///
(rarea ul ll mean_ARIA if SA == 0, col(orange%30) fintensity(inten80) lwidth(none)) ///
(line A mean_ARIA if SA == 0, col(orange)) ///
, graphregion(color(white)) ///
xtitle("mean ARIA score") ytitle("Predicted mean PDC at 12 months") ///
legend(order(4 "STEMI- All MI admissions" 2 "STEMI-Excluding peoiple with no dispensing") position(6
> ) ring(0) row(1) col(2) region(lcolor(white) color(none))) ///
yscale(range(0 1)) ylabel(0 "0" 0.1 "0.1" 0.2 "0.2" 0.3 "0.3" 0.4 "0.4" 0.5 "0.5" 0.6 "0.6" 0.7 "0.7
> " 0.8 "0.8" 0.9 "0.9" 1.0 "1.0" , angle(0) format(%9.0f)) xscale(range(0 5)) ///
title("`i´", placement(west) color(black) size(medium))
graph save "Graph" fig_`i´_0_SAcompare, replace
}
graph combine ///
fig_PDC_adj_P2Y12_1_SAcompare.gph ///
fig_PDC_adj_P2Y12_0SAcompare.gph ///
fig_PDC_adj_statins_1SAcompare.gph ///
fig_PDC_adj_statins_0SAcompare.gph ///
fig_PDC_adj_beta_1SAcompare.gph ///
fig_PDC_adj_beta_0SAcompare.gph ///
fig_PDC_adj_ACE_ARB_1SAcompare.gph ///
fig_PDC_adj_ACE_ARB_0SAcompare.gph ///
, altshrink cols(2) graphregion(color(white)) xsize(4.5)
graph export "G:\Adam\Project 2  - location and MI outcomes\Results\Fig_pdc_NSTEMI_STEMI_SAcompare.p
> df", as(pdf) name("Graph") replace
```

References

[1] D P Chew, I A Scott, L Cullen, J K French, T G Briffa, P A Tideman, S Woodruffe, A Kerr, M Branagan, P E Aylward, and Nhfa Csanz Acs Guideline Executive Working Group. National heart foundation of australia cardiac society of australia and new zealand: Australian clinical guidelines for the management of acute coronary syndromes 2016. *Heart Lung Circ*, 25:895–951, 2016. Chew, Derek P Scott, Ian A Cullen, Louise French, John K Briffa, Tom G Tideman, Philip A Woodruffe, Stephen Kerr, Alistair Branagan, Maree Aylward, Philip E G eng Letter Review Australia Heart Lung Circ. 2016 Sep;25(9):895-951. doi: 10.1016/j.hlc.2016.06.789. Epub 2016 Jun 16.

[2] Ariel Linden. *PDC: Stata module for computing the Proportion of Days Covered (of a medication)*. Statistical Software Components S458551, Boston College Department of Economics, 4 2019.

[3] David Prieto-Merino, Amy Mulick, Craig Armstrong, Helen Hoult, Scott Fawcett, Lina Eliasson, and Sarah Clifford. Estimating proportion of days covered (pdc) using real-world online medicine suppliers' datasets. *Journal of Pharmaceutical Policy and Practice*, 14, 12 2021.