IMPLANTABLE CARDIAC MONITORS TO DETECT ATRIAL FIBRILLATION AFTER CRYPTOGENIC STROKE (DAP42): QUALITY CHECK OF HEALTH ECONOMIC MODEL PROVIDED BY EXTERNAL ADVISORY GROUP

REPORT BY THE NICE DECISION SUPPORT UNIT

7th November 2019

Geoff Holmes, Sue E Ward, Allan J Wailoo

NICE Decision Support Unit,
ScHARR,
University of Sheffield
[redacted]

Tel [redacted]
E-mail [redacted]
Website [redacted]
Twitter [redacted]

ABOUT THE DECISION SUPPORT UNIT

The Decision Support Unit (DSU) is based at the University of Sheffield with members at York, Bristol, Leicester and the London School of Hygiene and Tropical Medicine. The DSU is commissioned by The National Institute for Health and Care Excellence (NICE) to provide a research and training resource to support the Institute's Centre for Health Technology Evaluation Programmes. Please see our website for further information www.nicedsu.org.uk

Acknowledgements

The production of this document was funded by the National Institute for Health and Care Excellence (NICE) through its Decision Support Unit. The views, and any errors or omissions, expressed in this document are of the authors only. NICE may take account of part or all of this document if it considers it appropriate, but it is not bound to do so.

1. EXECUTIVE SUMMARY

The Decision Support Unit (DSU) External Assessment Centre was asked by NICE to review the health economic model provided by an External advisory group (EAG). In particular, to identify whether there are any coding errors in the R component of the model which could significantly impact the model outputs; to fix any such errors identified and provide an updated model and updated results.

The DSU conducted an initial detailed review of the R code. This identified the need to substantially modify the R code, both to make it more transparent and to get it into a form amenable for input-output testing as described below. It also identified a few minor issues with the code described below, none of which are considered to have a major impact on results. One of these was a clear inconsistency (in gender weighting) which could be easily fixed and it resulted in a change of only £2 in the deterministic ICER of the overall model. Unfortunately, the associated PSA run aborted during execution and there wasn't time to re-run so this will be checked and reported later.

The task of extensive code modification was therefore undertaken next. The aim at this point was not to make substantive changes to what the code did but, as motivated above, to make it more accessible for further checking. The detail of changes made is provided later in this report but briefly it involved organising the code more logically, making it easier to inspect and modify key inputs to the model, simplifying code lines and removing redundancy, making the code more readable and efficient. Because the aim at this point was not to change the effect of the code, continual checking was carried out to make sure the substantive outputs of the model remained unchanged.

During the modification, two additions were also made to the code. Firstly, a record of the state occupancy was saved as this was needed for the input-output testing (see below). It also meant that state trace plots could be produced and these are presented for the base case model settings in Section 8 of this report. The state trace plots appear to be reasonable but this judgment should be verified by experts in the disease area. Secondly, an addition within the main code loop calculated the expected total number of strokes incident for each initial treatment. The results found that 172 strokes were avoided in the sub-population where atrial fibrillation was detected and 52 strokes were avoided in the whole population tested. The EAG used a different method to calculate the number of people who suffered one or more strokes. The EAG calculations (which have not been verified) found that stroke was avoided in 85 people in the AF detected population and in 25 people in the tested population. These figures are a corollary to the main model output and the number of strokes is already taken into account in the cost and QALY output of the model. However, to the extent that they add evidence to inform the committee's decision, these additional results may be taken into account.

Having completed the main modification of the code, a series of input-output tests was performed. These are described later in the report. The inputs to the model include various costs, health utility values, baseline hazards and hazard ratios for events such as stroke and

death. An input-output test means setting all or a subset of these inputs to specified values, predicting what effect these input setting should have on model behaviour and outputs and then running the model to see if the predicted behaviour is achieved. The important outputs in this sense are the total costs and QALYs and the history of how patients progress through the model (the state trace). These tests complement the visual checking of the code for errors by probing the functionality of the code for accuracy and consistency. In all the tests performed the predicted outcome was achieved.

With the code in its modified form, a second line-by-line check of the code was carried out. No errors or inconsistencies were identified during this stage. As this checking was being done, a description was created of what the code is doing as it progresses and a log was made of the changes made in the DSU modification of the code. Both this description and change log are provided in the main body of this report in order to facilitate any future checking and usage of the model.

A number of limitations of the DSU model checking should be noted. Firstly coding errors can be very difficult to identify and can be missed even by detailed readings of the code. The dual approach taken here of code checking and input-output testing lessen the chances of this but it would be unwise to ever fully guarantee that a code is error free. Secondly, an appended portion of the main R code script was an addition by the EAG. Due to an absence of description and comments in the code it was impossible to check this. It was, however, ascertained from the EAG that this code didn't feed into the key model outputs. Thirdly, the inputs to the R model are numerous and diverse. Some of them are generated in code which was not included with that provided for review. It was beyond the scope of the DSU's involvement to check all the inputs for appropriateness and accuracy of generation. Fourthly, the work required on the R model and the time available did not allow sufficient scope for review of the Excel component of the EAG model nor full detailed comparison with the Diamontopoulos Excel Model. Some exploratory investigation was done but this confirmed that more time would be needed to carry this out adequately.

Notwithstanding these limitations, the DSU analysis confirms that the R component of the EAG model is accurately coded and without any serious flaws that impact significantly on the overall economic modelling outcomes.

Contents

2. BRIEF DESCRIPTION OF R MODEL

The aim of the R model is to simulate the longer-term benefits for people whose atrial fibrillation is detected (who switch to anticoagulant treatment) and those whose atrial fibrillation is not detected (who remain on antiplatelet therapy or no treatment). The long term costs and QALYs from the R model serve as inputs for the short term Excel model developed by the EAG.

The R model is a discrete-time Markov multistate model with 3 month cycle length and lifetime horizon (100 years old). However, in any run of the model multiple samples are run. Each of these represents a different patient (or cohort) with associated time to progression from paroxysmal (asymptomatic) AF (Px) to persistent / permanent AF (Ps / Pm), which latter two are separately modelled but treated identically. Thus AF progression is modelled outside of the Markov model which itself models the progression through other significant events (see below) and treatment changes. Each sample also has associated, sampled characteristics such as baseline hazards, hazard ratios of various events etc. Some of these are sampled within the R-code, others are imported from files which have been created elsewhere, so that their accuracy code cannot be verified.

In the R model, patients begin on any one of 7 treatments as well as no treatment. Four of them are the direct oral anti-coagulants (DOACs): Apixaban (5mg bd), Dabigatran (150mg bd), Edoxaban (60mg od), Rivaroxaban (20mg od); also Coumarin (INR 2-3), Antiplatelet (>=150mg od), Antiplatelet (<150mg od). The Markovian progress of each of these is calculated independently and concurrently through each model run.

Only the results for six of these are outputted to Excel : Coumarin (INR 2-3), Apixaban (5mg bd), Dabigatran (150mg bd), Edoxaban (60mg od), Rivaroxaban (20mg od), Antiplatelet (<150mg od). These are respectively labelled with suffixes .war, .api, .dab, .edo, .riv, .asp

The model states are constructed from the basis of four important events: stroke (S), myocardial infarction (MI), major bleed (B) and intra-cranial haemorrhage (ICH). These can be experience in twos, threes and all four which together with no-event-experienced give a total of 16 states. These are multiplied up by the number of treatments (8) and finally death is added to give a total of 129 model states. There are three other events modelled which aren't included in states: Transient ischemic attack (TIA), systemic embolism (SE, these two are transient events) and Death; also the non-event 'Stay'.

3. BASE CASE MODEL OUTPUT

The updated R model produced output for export into the Excel model which is very similar to that produced by the original model. The only difference being due to aligning the gender weighting to be consistent throughout the model. This left the costs unchanged and changed the total QALYs for each treatment by around 0.12%, as calculated by the R model. In turn, this changed the deterministic discounted pairwise ICERs for the three devices by between £1 and £4 in the Excel model.

3.1 ICERs

The ICER results from the updated Excel model are shown in Tables 1-8. It should be noted that the probabilistic results are taken from the "Results - PSA!" worksheet of the Excel model as the live results on the "Results!" sheet only function correctly for the deterministic results.

Table 1 Base case pairwise cost effectiveness results (undiscounted)

Intervention	Total costs (£)	Total QALYs	Incremental costs (£)	Incremental QALYs (£)	ICER (£) vs. SoC
Standard of care	£11,049	2.14			
Reveal LINQ	£12,552	2.35	£1,503	0.21	£7,142
BioMonitor 2-AF	£11,782	2.35	£733	0.21	£3,482
Confirm RX	£12,317	2.28	£1,269	0.14	£8,886

Table 2 Base case pairwise cost effectiveness results (discounted)

Intervention	Total costs (£)	Total QALYs	Incremental costs (£)	Incremental QALYs (£)	ICER (£) vs. SoC
Standard of care	£7,600	1.74			
Reveal LINQ	£9,092	1.88	£1,492	0.14	£10,342
BioMonitor 2-AF	£8,322	1.88	£722	0.14	£5,006
Confirm RX	£8,866	1.84	£1,267	0.10	£12,879

Table 3 Probabilistic pairwise cost effectiveness results (undiscounted)

Intervention	Total costs (£)	Total QALYs	Incremental costs (£)	Incremental QALYs (£)	ICER (£) vs. SoC
Standard of care	£11,049	2.14			
Reveal LINQ	£12,553	2.35	£1,504	0.21	£7,149
BioMonitor 2-AF	£11,783	2.35	£734	0.21	£3,489
Confirm RX	£12,319	2.28	£1,270	0.14	£8,894

Table 4 Probabilistic pairwise cost effectiveness results (discounted)

Intervention	Total costs (£)	Total QALYs	Incremental costs (£)	Incremental QALYs (£)	ICER (£) vs. SoC
Standard of care	£7,600	1.74			
Reveal LINQ	£9,093	1.88	£1,493	0.14	£10,350
BioMonitor 2-AF	£8,323	1.88	£723	0.14	£5,014
Confirm RX	£8,867	1.84	£1,268	0.10	£12,888

Table 5 Base case incremental cost effectiveness results (undiscounted)

Intervention	Total costs (£)	Total QALYs	Incremental costs (£)	Incremental QALYs (£)	ICER (£) vs. SoC
Standard of care	£11,049	2.14			
BioMonitor AF-2	£11,782	2.35	£733	0.21	£3,482
Confirm RX	£12,317	2.28	£536	-0.07	Dominated
Reveal LINQ	£12,552	2.35	£234	0.00	Dominated

Table 6 Base case incremental cost effectiveness results (discounted)

Intervention	Total costs (£)	Total QALYs	Incremental costs (£)	Incremental QALYs (£)	ICER (£) vs. SoC
Standard of care	£7,600	1.74			
BioMonitor AF-2	£8,322	1.88	£722	0.14	£5,006
Confirm RX	£8,866	1.84	£544	-0.05	Dominated
Reveal LINQ	£9,092	1.88	£226	0.00	Dominated

Table 7 Probabilistic incremental cost effectiveness results (undiscounted)

Intervention	Total costs (£)	Total QALYs	Incremental costs (£)	Incremental QALYs (£)	ICER (£) vs. SoC
Standard of care	£11,049	2.14			
BioMonitor 2-AF	£11,783	2.35	£734	0.21	£3,489.42
Confirm RX	£12,319	2.28	£536	-0.07	Dominated
Reveal LINQ	£12,553	2.35	£770	0.00	Dominated

Table 8 Probabilistic incremental cost effectiveness results (discounted)

Intervention	Total costs (£)	Total QALYs	Incremental costs (£)	Incremental QALYs (£)	ICER (£) vs. SoC
Standard of care	£7,600	1.74			
BioMonitor 2-AF	£8,323	1.88	£723	0.14	£5,013.53
Confirm RX	£8,867	1.84	£544	-0.05	Dominated
Reveal LINQ	£9,093	1.88	£770	0.00	Dominated

3.2 CEACs

The cost-effectiveness acceptability curves for the three devices, with respect to standard of care are shown in Figures 1-3.

Figure 1 CEAC for Reveal LINQ with respect to standard of care

Figure 2 CEAC for Biomonitor AF-2 LINQ with respect to standard of care

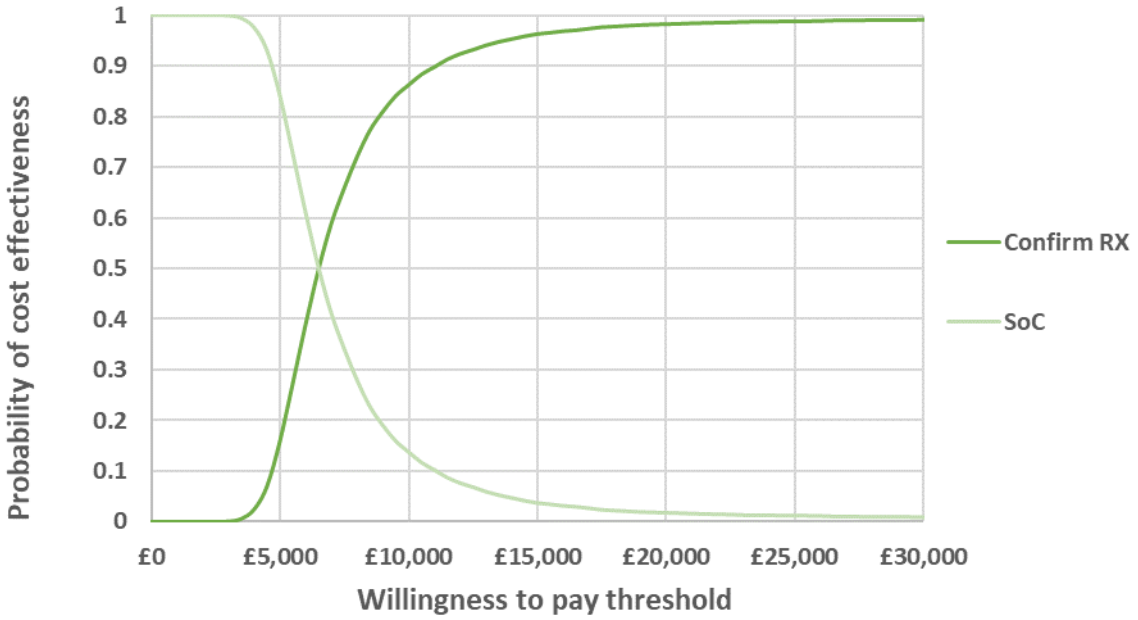


Figure 3 CEAC for Confirm RX with respect to standard of care

3.3 Number of strokes

An addition to the code in the main cycle loop, keeps a running total of new strokes. It does this by multiplying the state occupancy at each cycle by the state dependent probabilities of stroke and summing to get an expected value. It should be noted that this is the number of strokes not the number of people who have one or more strokes. The latter quantity is what was calculated by the EAG.

The number of strokes per 1,000 patients by initial DOAC treatment, averaged over 10,000 samples, is shown in Table 9.

The same methodology used by the EAG was then followed to calculate number of strokes avoided by the use of ICMs, as also shown in 9:

1. Get number strokes for ICM identified DOAC treated patients by calculating the sum, weighted by percentage of DOAC usage (as shown in column 3 of Table 9). Updated DOAC strokes, 193 (EAG had that 124 people had one or more strokes).
2. For standard care, it is assumed that 10% will be identified and transfer to DOACs the remaining 90% remaining on Aspirin. Updated SoC number of strokes, 366 (EAG had that 209 people had strokes). Therefore total strokes avoided using ICMs is 172.
3. The above figures relate to the 30% of the tested population who have AF detected. To get the number of strokes per 1,000 patients for the whole population, the figures are multiplied by 0.3 giving an updated figure of 52 strokes avoided.

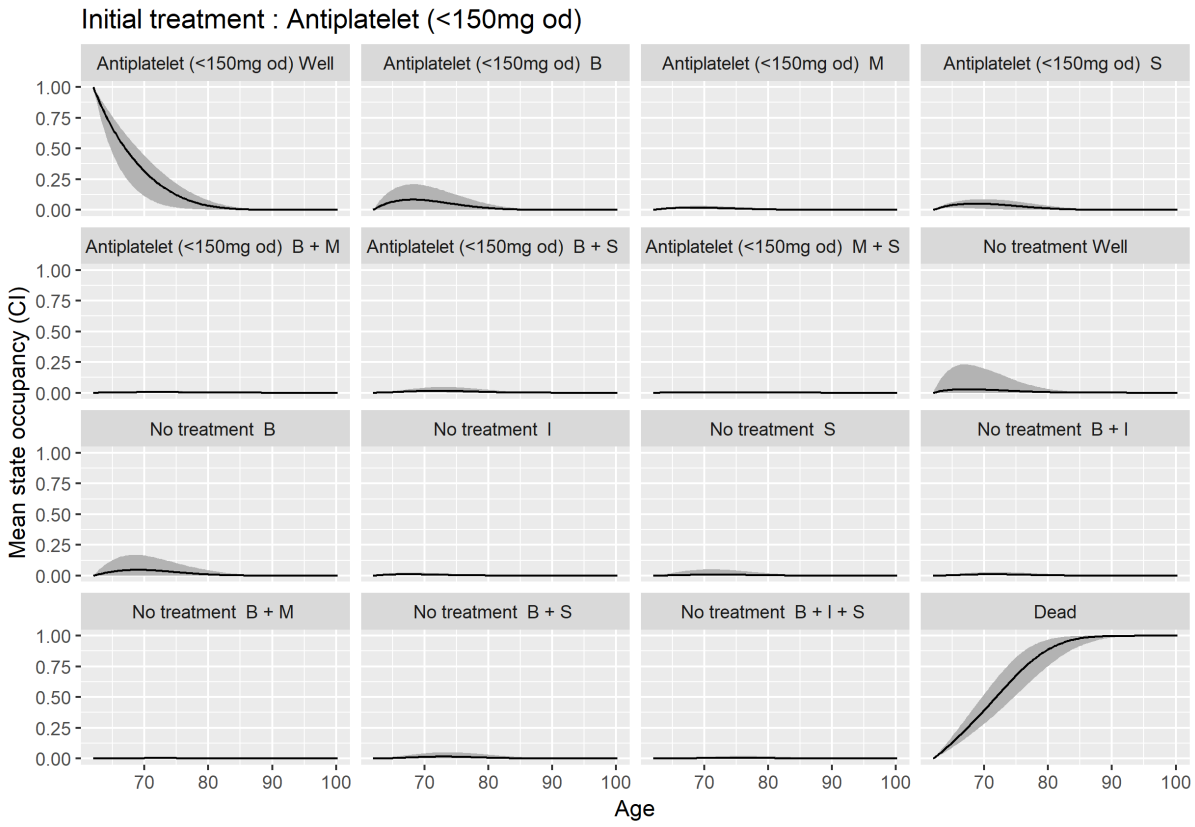
Table 9 Expected number of strokes per 1,000 patients and strokes avoided

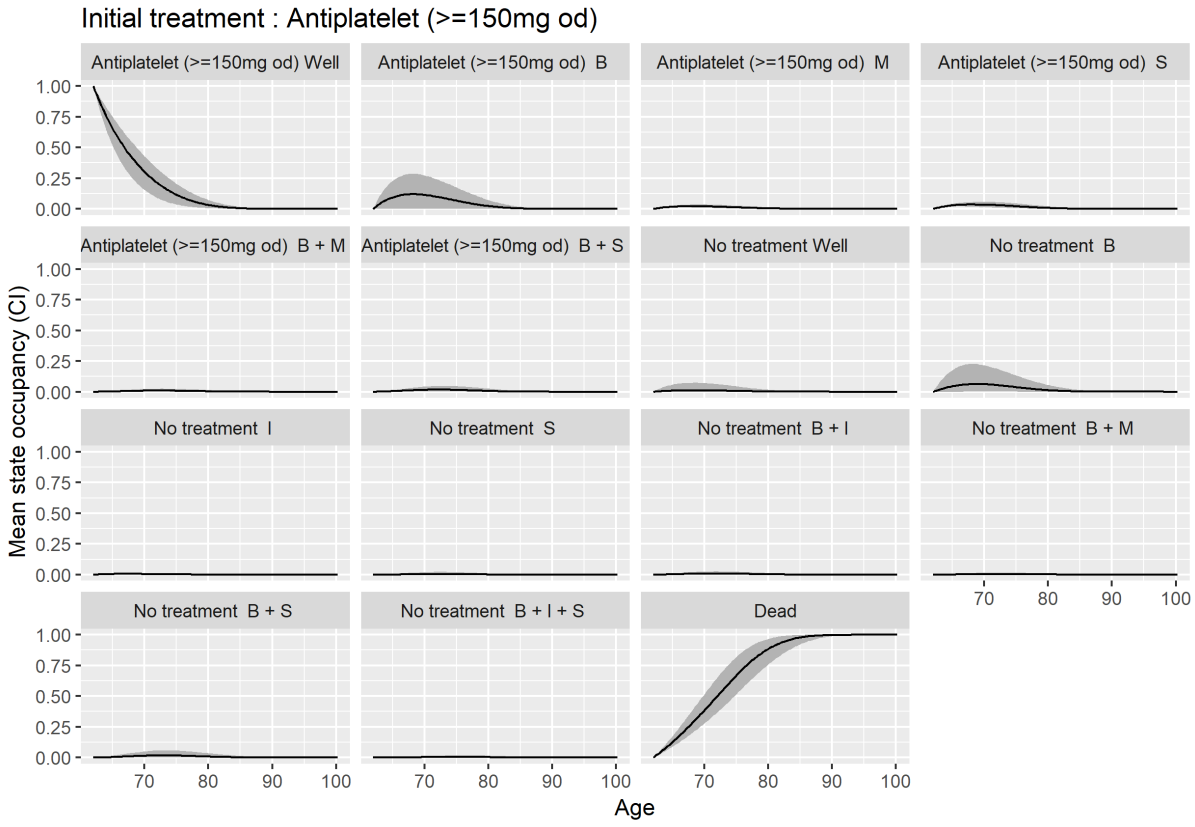
Treatment	Total strokes	DOAC % usage
Apixaban (5mg bd)	184	48.0
Dabigatran (150mg bd)	174	5.5
Edoxaban (60mg od)	197	2.9
Rivaroxaban (20mg od)	206	43.6
Aspirin	385	
ICM Weighted DOAC (identified sub-population)	193	
SoC (identified sub-population)	366	
Difference of above (strokes avoided)	172	
ICM Weighted DOAC (tested population)	58	
SoC (tested population)	110	
Difference of above (strokes avoided)	52	

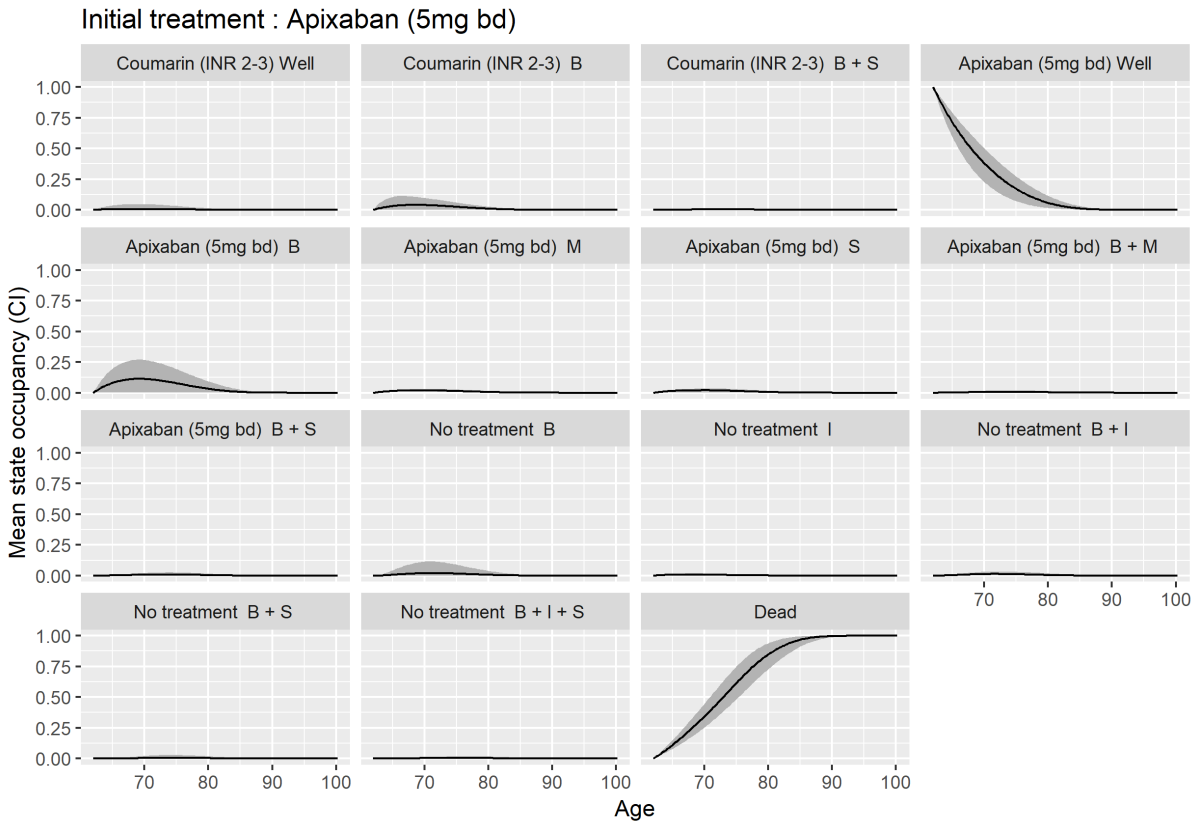
3.2 State trace plots

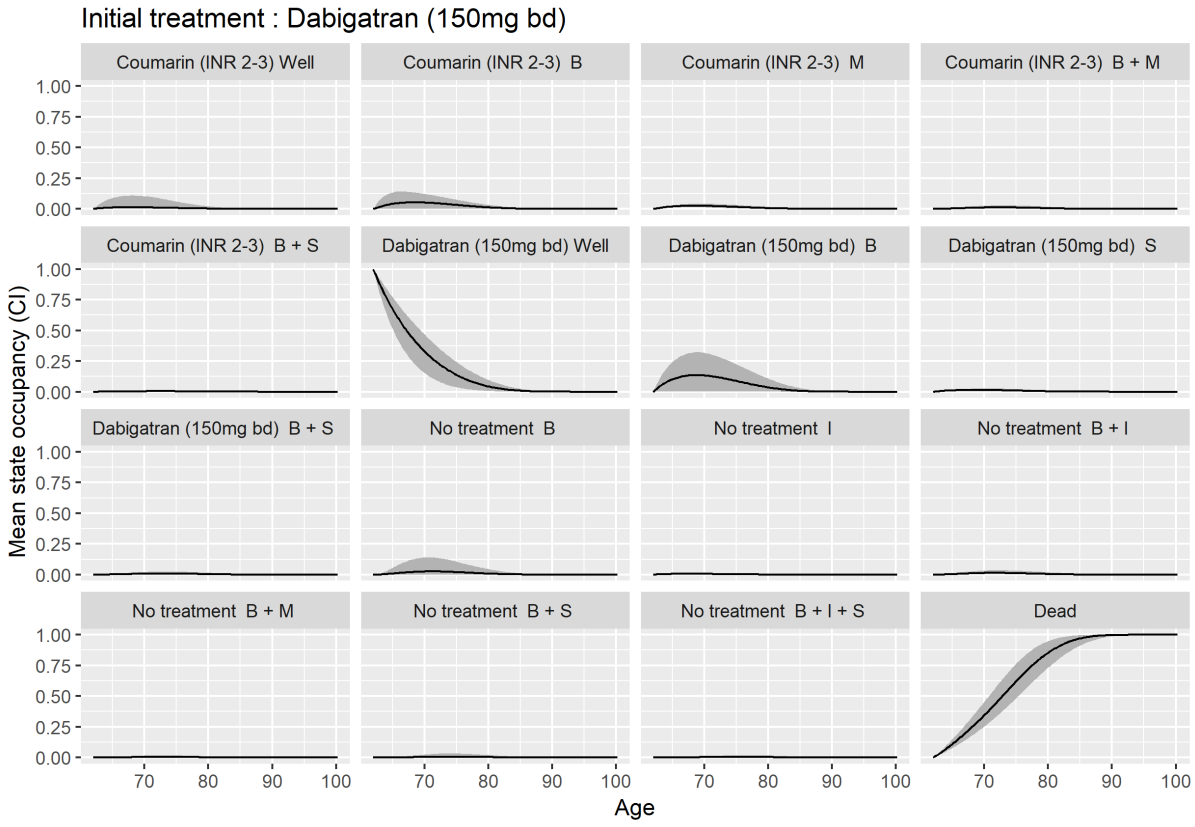
The following pages show the base case Markov state trace plots for each of the eight initial treatments in the R model. The solid lines show the mean state occupancy calculate over 10,000 samples and the grey region is the 95% confidence intervals at each cycle. Only those states that have non-negligible occupancy are shown for each initial treatment. Non-negligible here means that the average state occupancy over all cycles is greater than 1 in 1,000.

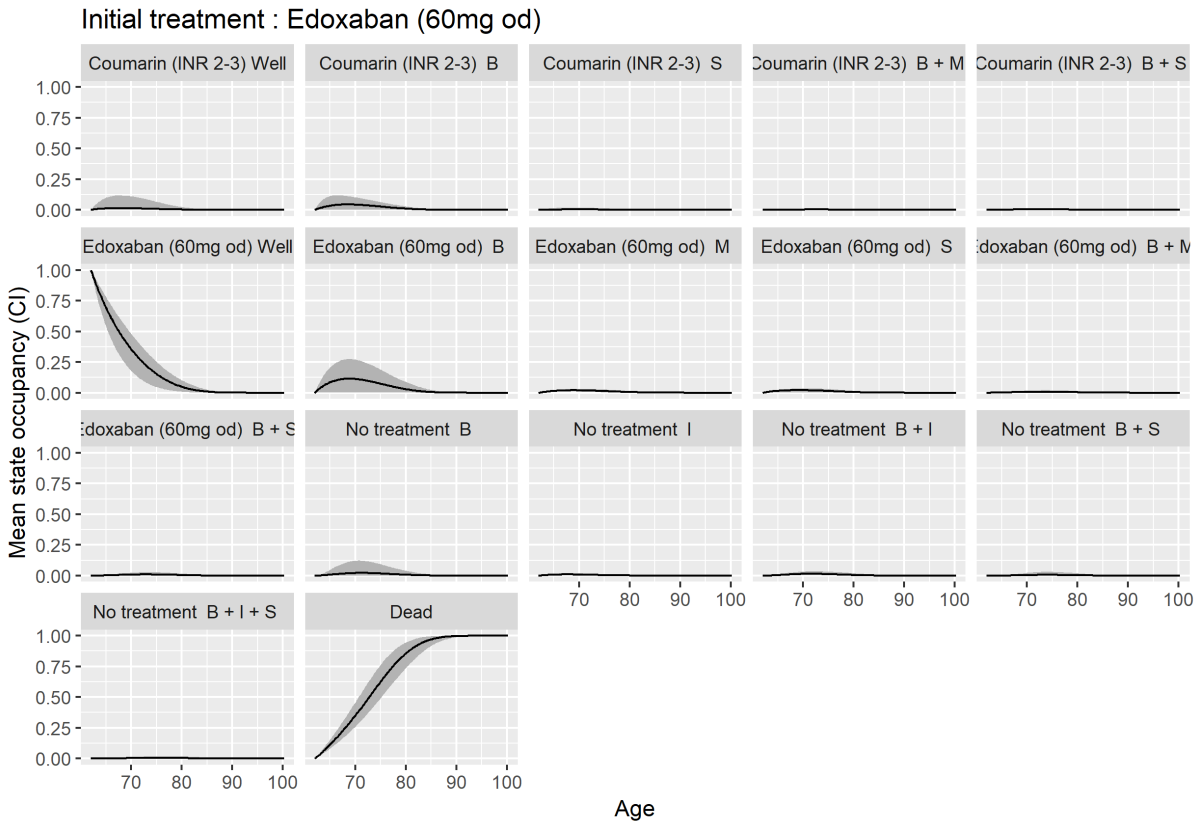
It is considered that these plots look reasonable but they should be verified by experts in the disease area.


Initial treatment : Antiplatelet (<150mg od)
Antiplatelet (<150mg od) Well
Antiplatelet (<150mg od) B
Antiplatelet (<150mg od) M
Antiplatelet (<150mg od) S
Antiplatelet (<150mg od) B + M
Antiplatelet (<150mg od) B + S
Antiplatelet (<150mg od) M + S
No treatment Well
No treatment B
No treatment I
No treatment S
No treatment B + I
No treatment B + M
No treatment B + S
No treatment B + I + S
Dead
Mean state occupancy (CI)
1.00
0.75
0.50
0.25
0.00
70
80
90
100
Age

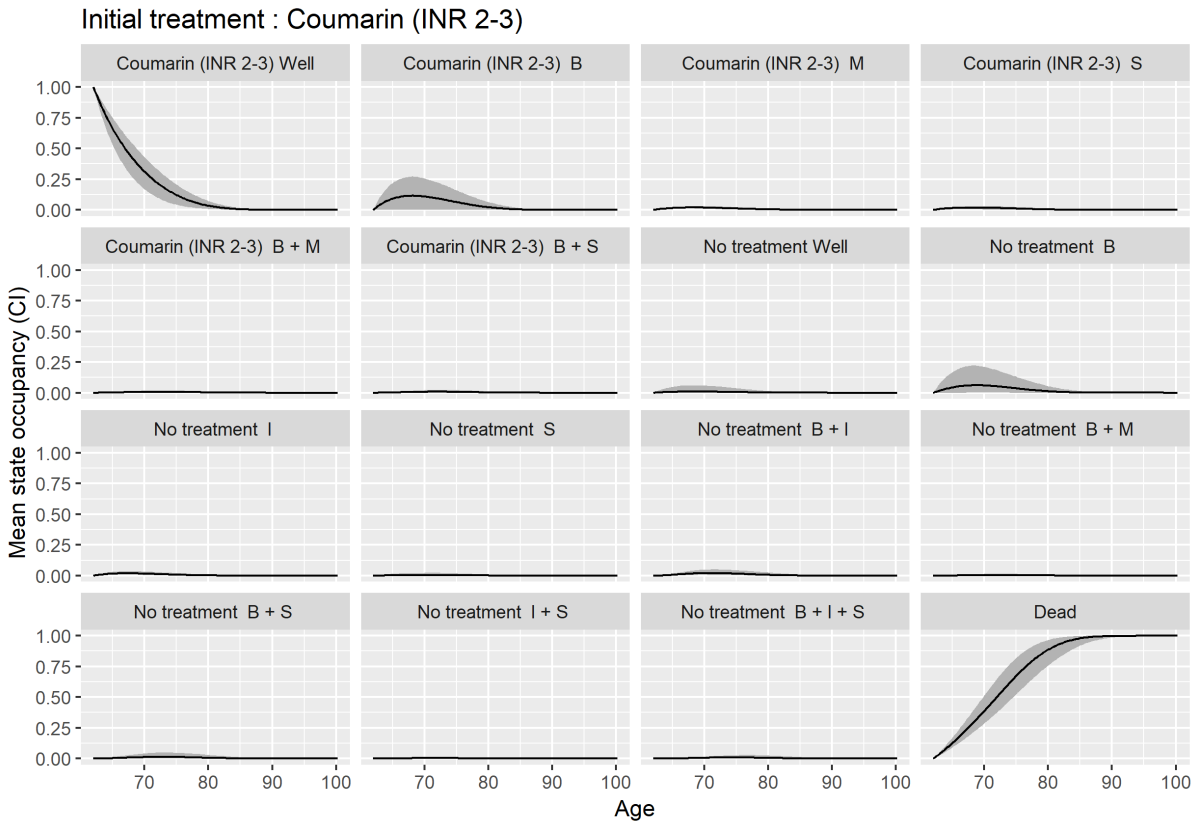

Initial treatment : Antiplatelet (>=150mg od)
Antiplatelet (>=150mg od) Well
Antiplatelet (>=150mg od) B
Antiplatelet (>=150mg od) M
Antiplatelet (>=150mg od) S
Antiplatelet (>=150mg od) B + M
Antiplatelet (>=150mg od) B + S
No treatment Well
No treatment B
No treatment I
No treatment S
No treatment B + I
No treatment B + M
No treatment B + S
No treatment B + I + S
Dead
Mean state occupancy (CI)
1.00
0.75
0.50
0.25
0.00
70
80
90
100
Age

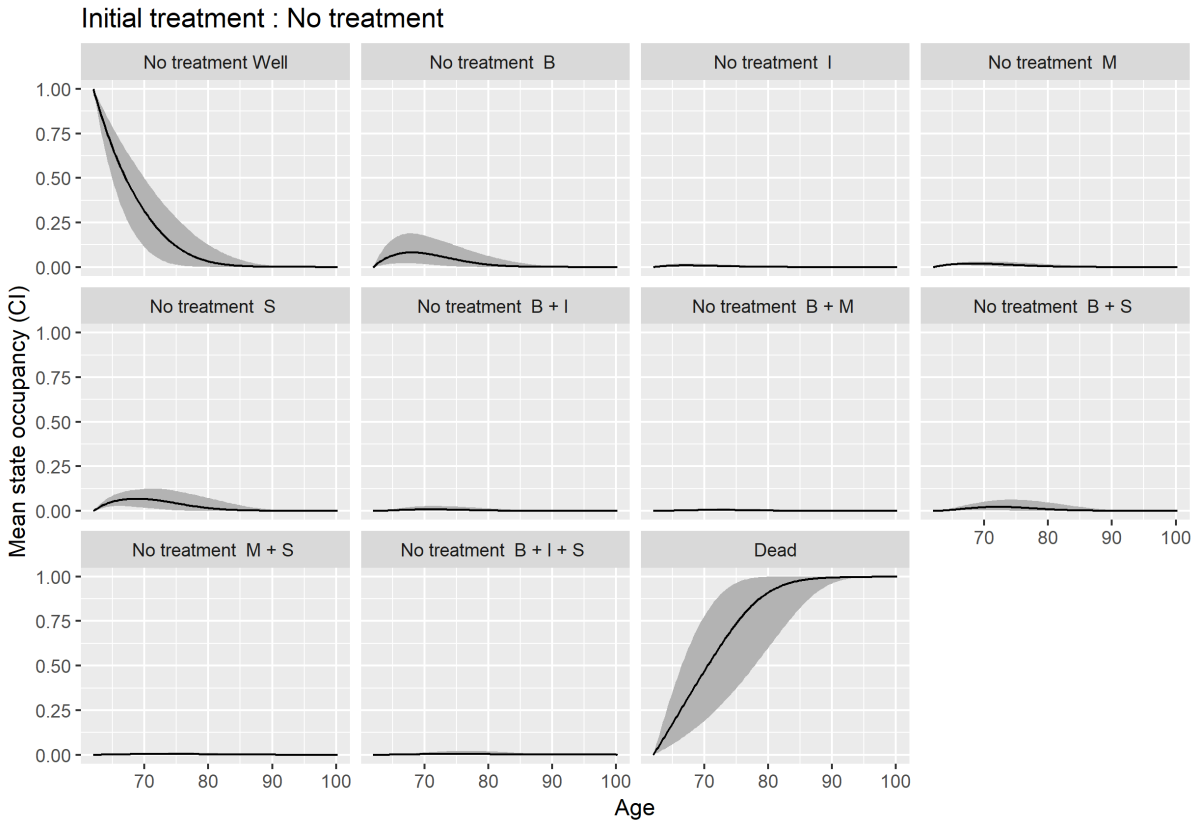

Initial treatment : Apixaban (5mg bd)
Coumarin (INR 2-3) Well
Coumarin (INR 2-3) B
Coumarin (INR 2-3) B + S
Apixaban (5mg bd) Well
Apixaban (5mg bd) B
Apixaban (5mg bd) M
Apixaban (5mg bd) S
Apixaban (5mg bd) B + M
Apixaban (5mg bd) B + S
No treatment B
No treatment I
No treatment B + I
No treatment B + S
No treatment B + I + S
Dead
Mean state occupancy (CI)
1.00
0.75
0.50
0.25
0.00
70
80
90
100
Age


Initial treatment : Dabigatran (150mg bd)
Coumarin (INR 2-3) Well
Coumarin (INR 2-3) B
Coumarin (INR 2-3) M
Coumarin (INR 2-3) B + M
Coumarin (INR 2-3) B + S
Dabigatran (150mg bd) Well
Dabigatran (150mg bd) B
Dabigatran (150mg bd) S
Dabigatran (150mg bd) B + S
No treatment B
No treatment I
No treatment B + I
No treatment B + M
No treatment B + S
No treatment B + I + S
Dead
Mean state occupancy (CI)
1.00
0.75
0.50
0.25
0.00
70
80
90
100
Age


Initial treatment : Edoxaban (60mg od)
Coumarin (INR 2-3) Well
Coumarin (INR 2-3) B
Coumarin (INR 2-3) S
Coumarin (INR 2-3) B + M
Coumarin (INR 2-3) B + S
Edoxaban (60mg od) Well
Edoxaban (60mg od) B
Edoxaban (60mg od) M
Edoxaban (60mg od) S
Edoxaban (60mg od) B + M
Edoxaban (60mg od) B + S
No treatment B
No treatment I
No treatment B + I
No treatment B + S
No treatment B + I + S
Dead
Mean state occupancy (CI)
1.00
0.75
0.50
0.25
0.00
70
80
90
100
Age


Initial treatment : Rivaroxaban (20mg od)
Coumarin (INR 2-3) Well
Coumarin (INR 2-3) B
Coumarin (INR 2-3) S
Coumarin (INR 2-3) B + M
Coumarin (INR 2-3) B + S
Rivaroxaban (20mg od) Well
Rivaroxaban (20mg od) B
Rivaroxaban (20mg od) M
Rivaroxaban (20mg od) S
Rivaroxaban (20mg od) B + M
Rivaroxaban (20mg od) B + S
No treatment B
No treatment I
No treatment B + I
No treatment B + S
No treatment B + I + S
Dead
Mean state occupancy (CI)
1.00
0.75
0.50
0.25
0.00
70
80
90
100
Age


Initial treatment : Coumarin (INR 2-3)
Coumarin (INR 2-3) Well
Coumarin (INR 2-3) B
Coumarin (INR 2-3) M
Coumarin (INR 2-3) S
Coumarin (INR 2-3) B + M
Coumarin (INR 2-3) B + S
No treatment Well
No treatment B
No treatment I
No treatment S
No treatment B + I
No treatment B + M
No treatment B + S
No treatment I + S
No treatment B + I + S
Dead
Mean state occupancy (CI)
1.00
0.75
0.50
0.25
0.00
70
80
90
100
Age


Initial treatment : No treatment
No treatment Well
No treatment B
No treatment I
No treatment M
No treatment S
No treatment B + I
No treatment B + M
No treatment B + S
No treatment M + S
No treatment B + I + S
Dead
Mean state occupancy (CI)
1.00
0.75
0.50
0.25
0.00
70
80
90
100
Age

4. MODEL INPUT-OUTPUT TESTING

A series of input-output tests was conducted to probe whether aspects of the code have their desired effect as shown in Table 10

Table 10 input-output tests with defined expected outcome

Test	Description	Expected outcome	Pass / Fail
0	Delete relevant outputs and re-run original code	Output created in unchanged form	Pass
1	Set all costs to zero (n.samples=2, initial.age=90)	Total costs zero for all treatments	Pass
2	Set all utilities and utility decrements to zero (n.samples=2, initial.age=90)	Total QALYs zero for all treatments	Pass
3	Set baseline death rate to zero (n.samples=2, initial.age=90)	Death state always empty	Pass
4	Set baseline death rate to unity, HR for death to 1 and all other event rates zero (n.samples=2, initial.age=90)	Death state always full	Pass
5	Set all treatment switch probabilities to zero (n.samples=10, initial.age=90)	No switching (having checked that switching does occur for these settings in base case)	Pass
6	Set all treatment switch probabilities to zero and all event hazards to zero (n.samples=10, initial.age=90)	All remain well and in initial treatment, tested via: sum(cohort.trace[,,,1]>0)==8*41	Pass
7	As 6 but baseline hazard of death is 1	All go from initial treatment well to Dead	Pass
8	As 6 but with bleed hazard of 1 and bleed switch probability of 1	Coumarin-well to bleed + no-treatment; 4 DOACs-well to bleed+no-treatment via bleed+Coumarin ; Anti-platelets-all to bleed+no-treatment; No-treatment-well to bleed+no-treatment	Pass
9	As 8 but for Stroke instead of bleed	Treatment switching as 8 but replace bleed by stroke in description of outcome	Pass
10	As 9 but all HRs set to 1 (including event and treatment HRs) also n.samples=1000	All 4 DOAC curves to be identical and all 3 non-DOAC curves to be identical	Pass
11	As 10 but for bleed instead of stroke	All 4 DOAC curves to be identical and all 3 non-DOAC curves to be identical	Pass
12	Set all baseline hazards to 0 except Stroke as base case; all HRs to 1 except HRs for stroke by treatment set 2:7	Number of strokes to follow same pattern and be in exact proportions 2:7 after first cycle	Pass
13	Set all baseline hazards to 0 except Stroke and Bleed both 0.1, set HR for stroke due to past stroke to 2; turn off strokes for api, turn off bleeds for dab	Expect more dab strokes than api bleeds	Pass

5. General comments on the original R code

a) The code has had several earlier incarnations. However, there was no record of this within the code which would have been helpful.
b) Inputs to the model appeared in numerous different places sometimes with little or no descriptive comment. Some were hard coded within sub-functions, some were imported from other files whose origin was unclear and their accuracy could not therefore be verified.
c) There was a significant level of redundancy (variables defined but not ultimately used, data imported and not used or else subsequently overwritten).
d) May code lines were longer than can be read without right scrolling the editor, which is bad coding practice.
e) Indexing of array was often done in a very verbose way. This and other similar practices made for opaque coding.
f) The implementation of the model itself was complex without a clear description of how the various parts are intended to work together.
g) Part of the code added by the EAG (original lines 179 onwards) dealing with incidences was coded in such a way that it was not possible to check.

6. MINOR POINTS OF CONCERN IN THE ORIGINAL CODE

Not all of these have been checked for impact on the results but it is not expected that any of them will have a major effect.

a) In generate.transition.matrix there was a typo in the line:
if(sensitivity=="HRLowerAymptomatic")
This meant the associated sensitivity code would not have run when intended. This wouldn't affect the base case results.
b) In calculating treatment switch probabilities (age.indep.gen.probs.R) comment says "Only switch after MI if on Dabigatran", but this doesn't seem to be reflected when the treatment switching indices are applied in generate.transition.matrix function.
c) The life table data doesn't agree with what appeared to be the equivalent data from ONS (neither UK nor England / Wales).
d) In generate.hr.death.age function it would possibly be better to do weighted sum of hazards rather than weighted sum of probabilities, but given the small values involved it makes no difference.
e) In kind.age.factor.R function:
    - Shape1 parameter for age 65 doesn't fit the pattern of otherwise decreasing values.
    - The resulting age utility factors are not necessarily decreasing and are not so in some samples.
    - Gender weighting was 60/40 rather than 65/35 as in generate.hr.death.age (now both aligned to 65/35, changing the total QALYs by only around 0.12% for each of the treatments.)

7. OVERVIEW OF TASKS COMPLETED

a) A git version control repository was created to track all changes. This is a hidden folder which sits alongside the code and which keeps an ongoing log of what changes are made and why. It allows immediate reversion back to earlier or original versions of code for comparison or if an error has been introduced. This repository can be provided if needed.

b) The code was worked through in detail, line-by-line, to understand the general approach to the model and to spot potential conflicts with described accounts of the model (see description in later section).

c) Revisions of the code:
   - Some functions rewritten or edited to make them more efficient and easier to follow (and checked for consistency against original versions).
   - The main code has been converted to a function, and likewise some other sourced scripts. As an interim state, all variables required within functions were passed as parameters for checking of consistency. This was later revised to prevent memory overload and keep things simpler.
   - Collection together of facets of the code such as user settings, model inputs, initialisation of internal variables so they can easily be inspected and tested.
   - Removed obsolete code, variables, file inputs, unnecessary outputs.
   - Instead of saving cumulative costs and QALYs and then differencing at the end to get the by-cycle quantities, these are stored initially. Cumulative totals are still stored / unaffected.

d) Additions to the code:
   - Where possible inputs have been collected in an excel file where they can be edited for sanity checking and unit testing. This file is loaded and unpacked in R, so that mostly it is not necessary to have important inputs hard coded / hidden in the R code.
   - Store a Markov trace for one of the samples run through the model with option to plot the states which have non-zero occupancy.
   - Compute the expected number of strokes for each treatment at each cycle and keep a running total.
   - The primary initial-treatment-dependent outputs can now be saved as sheets in a single .xlsx file for quicker import into the Excel model (however, it was later found that this was potentially unstable for runs with n.samples of 1000 and above.

NB: at each stage of editing and adding to the code results were run and checked and it was found that existing results were reproduced i.e. no substantive changes have been made in terms of outputs.

8. DETAILED DESCRIPTION OF HOW THE UPDATED CODE WORKS

8.1 Running Main_DOAC_model_code.R does the following:

1. Clear all objects from workspace, console window and graphics objects
2. Install / load required packages
3. Source code from other scripts that create sub-functions
4. Create main DOAC.model function

The main function can now be called with various optional user settings which are passed as parameters e.g.:

```
DOAC.model()
```

or

```
DOAC.model(n.samples=10000)
```

The full set of default settings are:

```
DOAC.model<-function(
                n.samples=100,
                sensitivity="NULL",
                contra=TRUE,
                initial.age=62,
                initial.af.type="Px",
                random.seed=10,
                lifetables.file="Model inputs/EnglandWales-LifeTables-2015-17.csv",
                save.xlsx.output=F,
                save.indiv.csvs=T,
                store.trace=T,
                trace.ci.alpha=0.05,
                plot.markov.traces=FALSE,
                deleteGlobalVars=TRUE
)
```

Some of these parameters relate to variables set in the body of the original code, others are new. Description of all of them can be found in lines 56-69 of the main "Main_DOAC_model_code.R"

8.2 Calling the function as above does the following:

1. (Lines 71-169)
   Set random seed (for repeatability) and create many useful objects (treatment names vector, event names vector, etc) as global variables.
   Define treatment switch indices, i.e. how to map the state change related to a treatment change. These are defined to map the rules in the conceptual model.
2. (Lines 170-223) Initialise arrays which will store the outputs etc.
3. (Lines 224-342) Load and process model inputs (which are all now put in the "Model inputs" folder)
   a. grh_input_file.xlsx

This file collects together many inputs that were previously hard coded within the model code, allowing them to be more easily altered for testing: Distribution parameters for Costs, Utilities, Treatment switch probabilities, Hazard rate for stroke/mortality. It also allows the setting of values that override other inputs loaded separately into the model (see below): Baseline log event hazards, hazard ratio for Ps versus Px, rate of progression Px to Ps, management costs for initiation and maintenance, age utility factors.

b. Inputs.coda.csv[1]
   Two fields are used from this file: ratePxtochronic (for AF progression) and HR_Ps_v_Px (hazard ratio of stroke / mortality in Ps AF versus Px AF). There are multiple samples for each of these values. If selected in grh_input_file, all values for either can be overwritten by a single value for testing.
c. grh_getcosts_utils() is a function (based on code previously in a non-function script) which uses the inputs imported from grh_inputs_file.xlsx and management costs from the file "Model costs.xlsx[2]" to sample values for state costs, state management costs (initiation and maintenance), state utilities and event (dis)utilities.
d. get.utility.factor() is a function to get samples of age utility factors relative to age 65. The input parameters controlling this are hard coded in the function but the calculated values can be overridden by single values for each age if chosen in grh_input_file.xlsx.
e. bugs.baseline.csv[3]
   This loads samples of baseline hazards for each event. If selected in grh_input_file, all values for any event can be overwritten by a single value for testing.
f. grh_hr_overrides.csv
   This file contains further override values for the inputs in bugs.loghr.csv, hr.no.treatment.rda, hr.event.history() which are described below.
g. bugs.loghr.csv[4]
   This loads samples of hazard ratios for each event due to each treatment. If selected in grh_hr_overrides.csv, all values for any HR can be overwritten by a single value for testing.
h. hr.no.treatment.rda[5]
   Loading this creates a table, hr.no.treament, containing samples of hazard ratios for each event representing the efficacy of warfarin compared to no treatment.

---

[1] EAG: These came with the model and were not adapted by us. If you look at Table 32 on page 96 of the screening model HTA (https://www.journalslibrary.nihr.ac.uk/hta/hta21290#/abstract) I think the "HR_Ps_v_Px" values represent the pooled estimate referred to at the bottom of the table. Also, on page 106, Table 40 lists the model inputs and makes reference to the values for ratePxtochronic. The mean of the values I have in the model files match those values reported in the paper.

[2] Although this has sheets for Tests, Consumables, Staff, these don't appear to be used and hard values on the AF management cost sheet are inflated using the inflation sheet. EAG: The costs that aren't used were used for the AF screening model but that's the only reason they are in there. We haven't adapted these costs. See Table 39 on page 104 of the screening model HTA listed above.

[3] EAG: See screening HTA

[4] EAG: See screening HTA. The NMA to estimate treatment effects I believe was done by Nicky Welton but we were only in contact with Howard

[5] EAG: See screening HTA

The HRs for MI in this table are set manually to 1 (citing no evidence for a difference). If selected in grh_hr_overrides.csv, all values for any HR can be overwritten by a single value for testing.

i. hr.death.age() is a function which loads a lifetables file and creates a hazard ratio of death for each age modelled.
j. age.independent.samples() is a function which uses the inputs loaded from inputs.coda.csv and grh_inputs_file.csv (described above) to create samples of: cycles to AF progression, hazard ratio for stroke (and TIA/SE) due to Ps and Pm AF versus Px AF; hazard ratio for stroke and mortality for Ps versus Px AF; event costs; treatment switch probabilities due to events. Also set are samples of stroke.init.probs, the probability that a patient will initiate treatment (versus Apixaban) if they have a stroke. If the user setting, CONTRA==TRUE, in main function call, all samples are set to zero (corresponding to "has had bleed") which is the baseline setting. Otherwise all sample are set to 1.
hr.event.history() is a function that samples hazard ratios of events due to previous events (i.e. the effect that a bleed has on the hazard of future MI etc). The distribution parameters for these are loaded within the function from the file hr.event.history.xlsx. If selected in grh_hr_overrides.csv, all values for any HR can be overwritten by a single value for testing.

4. (Lines 343-448) Main loop over cycles (each being ¼ year)
   k. Determine the current AF type of each sample (compare cycle number to cycle to progress).
   l. Every four cycles, update the transition matrix (see generate.probabilities() below), transient utility and transient cost (i.e. those dis-utilities and costs connected with events) objects.
   m. Increment total costs by adding new state costs, state management costs and transient costs.
   n. Increment total QALYs by adding state utilities and transient utilities both scaled by the relevant age utility factor.
   o. Use the transition matrix to update the cohort.
   p. For each treatment, store the cost incurred and QALY gained during this cycle.
   q. If required by user setting, store summary stats of cohort vector.
5. (Lines 450-537) Post processing
   r. Save outputs as required in user settings
   s. Make state trace plots if required in user settings
   t. Return cohort trance summaries, total costs, total QALYs, sampled costs, QALYs and number of strokes for each treatment, and any state trace plots.

8.2 generate.probabilities() – function created in generate.transition.matrix.R

1. This is called every four cycles to define the live version of the time-varying Markov transition matrix, and the transient costs and utilities related to events that may happen in the forthcoming 4 cycles. These are all dependent on current age and current AF type in the cohort.
2. (Lines 30-98) Initialise and then build probability matrix. The element of this matrix in the *i*th row and *j*th column will be the probability of having the *i*th event when in the *j*th (non-death) state, where a state is defined by the current treatment and the event history (8*16=128 non-death states).

a. Loop over the 8 treatments
    i. Lookup up HR of stroke/SE/TIA for current AF type.
    ii. Loop over the 16 health states (i.e. without treatment dimension = event histories). So what follows is a calculation of hazards leading to probabilities of the events for one of the 128 non death states.
        1. Start with baseline hazard of each event (from bugs.baseline)
        2. Apply HR of stroke/SE/TIA due to AF type, to those hazards.
        3. If user option, sensitivity=`HRLowerAsymptomatic', divide hazard of Stroke and death by the symptomatic HR.
        4. Apply HR for death due to age (calculated from life-table data) to the hazard for death.
        5. Apply HR for all events due to treatment / no-treatment (relative to Warfarin).
        6. Apply HR for all events due to previous events incurred in the definition of the state.
        7. The hazard of each event for the state in question is now defined and these hazards are converted to event probabilities using a competing risk calculation. This completes the probability matrix for this state.

3. (Lines 102-107) Calculate the expected transient costs and utilities, i.e. those associated with events, by multiplying the probability matrix by the event costs and event utilities respectively.
4. (Lines 112-268) Initialise and build the transition matrix from the probability matrix.
    a. Loop over six state-defining treatments and loop over sixteen event history states. Each iteration of the inner loop represents one state which is the 'old' state being transitioned from in the transition matrix.
        i. Get the name and index of the old state (i.e. fix on the one appropriate row of the transition matrix to update in this iteration of the loop)
        ii. For each of the four state changing events (B,I,M,S = Bleed, ICH, MI, Stroke) and over the probabilities of no-switch / treatment-switch, add the corresponding probabilities to the appropriate columns for this row. NB the probabilities have to be added because multiple events will map the old state back to itself (i.e. B+S is mapped to itself by both B and S). The calculations for B,I,M are straightforward, that for S (Stroke) is more complex to allow for treatment discontinuation (if history of Bleed or ICH or on NOAC) or treatment initiation (no history of Bleed or ICH and `probably' not on NOAC).
        iii. Probability of death is straightforwardly added
    b. Add Death to Death transition probability as one.
5. Return transition matrix, transient costs, transient utilities.

9. Detailed description of changes to the code

9.1 High level changes

1. Some folders and files were unused and so were deleted to simplify and prevent confusion:
    a. Folder TTD/
    b. objects.Rdata
    c. Folder Bugs objects/ (items from this folder were loaded but then not used)
    d. Folder Model code/data (LifeTables file move to folder Model inputs/)
2. The main code has been converted to a function.
3. Other sourced scripts have been either made into functions or else the code has been brought into the main function.
4. Input parameters hard coded within the model have been collected together and moved to a separate file for easier inspection and so they can be altered for testing.
5. There were quite a few variables within the code that were defined but then not used so these were deleted e.g:
    a. k.treat
    b. t.names
    c. new.state.indices
    d. cohort.on.trt
    e. event.codes
6. Collection together of facets of the code such as user settings, model inputs, initialisation of internal variables so they can easily be inspected and tested.
7. Throughout the code, lines that overflow a normal window in RStudio have been shortened to a more standard width, spreading over multiple lines as necessary.

9.2 Low level changes

(Key: OC – original code, UC – updated code,
OL - original line numbers, UL – updated line numbers)

1. Main_DOAC_model.code.R (renamed with underscores)
    a. The updated code (UC) does not change directories at any point (this is unnecessary and causes frustration if the model is stopped or crashed mid run) so old lines (OL) such as OL 11,28,31 etc not needed. The sourced files (OL 42-47) are sourced outside the main function in UC updated lines (UL) 25-30, where they now don't have any code other than that to create sub-function that are called within the main function.
    b. User settings hard-coded in original code (OC) such as in OL 13-15, 33-37,49 are passed as function parameters in the UC, updated lines (UL) 36-70.
    c. Everything considered a model input is dealt with in a re-configured code section for this: OL 17-23,63, are gathered with other inputs in UL 224-339. This inputs section in UC also contains much new code to load parameters which were hard-coded in OC but are now in the external file grh_input_file.xlsx; and also code for overriding the base case inputs parameters which was needed for the unit testing.
    d. OL 63, age.independent.samples is unpacked into separate variables UL316-325 to make later use less verbose (compare OL 84-91 to UL 357-374).

e. All code which represents initialisation is collection together in a newly configured code section. i.e. OL 13,40,52-54,58-74 are gathered with all initialisation in UL 170-221. This also contains code for storing statistics of the cohort traces and for storing the number of new strokes at each cycle.
f. OL97-100, because the values unpacked from the new version of age.independent.samples (see d above) are made global variables, the function call can be much simpler.
g. OL 103, commenting this line out (UL 385) made no difference
h. UC 392-396, addition to log number of new strokes at each cycle
i. OC 108-135, the for loop over samples has been 'vectorised' using lapply function (UC 402-431). This hope was that this would speed things up, however, that is not established as yet. Also, in this section, instead of incrementing the total costs, a separate newcosts variable is used (also for qalys). This avoids having to do the rather odd backtracking over the outputs in OL 138-151.
j. OL 175 onwards didn't add anything relevant to outputs required for the Excel model. Furthermore, they were not easy to understand and contained no comments or explanation so it was impossible to review them. These lines are deleted in the UC.
k. UL 475-495 saves the output needed for Excel as a .xlsx file. This idea was that this would make is quicker to copy into the Excel model (1 step rather than 12 steps) but it seems to cause crashes due to memory overload when using other than small values of n.sample (crashed when n.samle=1000).
l. UL 505-537, because this Main code is now a function relevant outputs are return to the user.

2. Model code/utility.function.R – the two functions in here weren't used at all within the code so they have been deleted. Instead this script creates four functions:
    a. plot.markov.state.traces – a completely new function
    b. next.state.name – this is an adapted version of the function with the same name in the OC. It uses a lookup table to find the next state much more succinctly. The lookup table (state.transition.map) is defined in UL 162-167 of Main_DOAC_model_code.R. It is perhaps a little less transparent that the original but it has been checked against the original to confirm that output is identical.
    c. generate.hr.death.age and generate.hr.death.age.old (formerly in its own script). The former is the updated version of the latter. Not much difference and same output but with slightly simpler indexing. The former is used and the latter is redundant.
3. Model code/generate.transition.matrix.R
    a. OL 1-87, is all preamble to the definition of the function. It was decided that it was more simple and transparent to include this in the relevant sections of the main function defined in Main_DOAC_model_code.R: setup (UL71-167), initialisation (UL 170-221) and inputs (UL 224-339).
    b. OL 89-183 which define the function age.independent.generate.probabilites have been removed and moved to a separate file Model code/age.indep.gen.probs.old.R. This has been further edited as discussed below.

c. OL 191, only age, af.type change from cycle to cycle so the rest of the parameters (having been made globals) are not passed (apart from sensitivity) cd UL 19-23.
d. Whilst this function can take a vector of ages (the original default was to calculate for all ages at once), in the OC only one age was passed at a time – when the function was called in each cycle. Thus the for loop over ages (OL 95) was redundant and has been removed for clarity.
e. OL 206, k.treat was unused and removed.
f. OL 209-210, in UC bugs.baseline has already been reduced to n.samples
g. OL 214-219, type.hr is independent of i.health.state so has been moved outside this loop and simplified (UC 33-35). Possibly it could also have been moved outside the treatments loop.
h. OL 220-222, three separate calculations done in one, UL 52-53.
i. OL 225, a typo here, corrected in UL 56, means that the condition would never have been true and hence the associated sensitivity test result would not have been valid.
j. OL 260-275, as mentioned above, age indexing was not necessary and its removal simplifies the code, UL 95-112.
k. OL289-351, the logic here is unchanged but by removing the redundant indexing over age, using shortcuts for some of the indexing and reducing all lines to a more standard length, the code becomes a lot more accessible.

4. Model code/age.indep.gen.probs.old.R (see 3.b) is superseded by Model code/age.indep.gen.probs.R
    a. OL 13 n.samples already extracted so not needed in UL 19.
    b. All hard coded inputs are now referenced from the inputs variable which is derived from the file grh_input_file.csv
    c. OL97-98 bugs.baseline and bugs.loghr are dealt with in the inputs section of the main code.
5. Model code/generate.costs.utilities.R
    a. This script now creates a function which is called within the new main function (within the inputs sections).
    b. OL 8-14 etc, all hard-coded input parameters are now pulled in from a single file, grh_input_file.xlsx which is then unpacked (in the inputs section of the main function) into the inputs variable, as referenced in UL 11-17.
    c. OL 21, Model costs.xlsx is now within the Model inputs/ folder.
    d. In the UC, there are now extra lines to override the base case inputs if required for testing (e.g. UL 24-25, 39-40 etc).
    e. OL 84-115, there wasn't time to simplify these lines unfortunately.
6. Model code/sample.hr.event.history.R
    a. Defines a function called in the main function as part of the inputs setup.
    b. OL 7 not needed as the package is loaded in the main script.
    c. OL 9-17 have been moved inside the function. The function is only called once so this creates no overhead and has the advantage of extra items not persisting in the workspace when no-longer needed.
    d. The loaded file hr.event.history.xlsx has been moved from Model code/data/ to Model inputs/.

e. In the UC there are two version of the function. One is based entirely on the OC (sample.hr.event.history), the other (sample.hr.event.history.grh) is much more concise. However, the new version changes the order of sampling from the random number generator so alters the actual results for a given random seed. The former version is therefore the one still in use – to facilitate easier checking that nothing substantive has been changed but the code updates.

7. kind.age.factor.R – the only change to this file, is that the gender weightings have been made consistent with those used in generate.hr.death.age (65:35 as specified by EAG versus 60:40 in the model they inherited). It is the only remaining place where some inputs are hard-coded (those for the beta distributions for age utility factors).

NB: at each stage of editing and adding to the code (up until that made in 7 above), results were run and checked against the .csv output provided with the model (n.samples=100, set.seed(10)) and it was found that existing results were reproduced exactly i.e. no substantive changes have been made in terms of outputs.

APPENDIX : R AND PACKAGE VERSIONS USED FOR DEVELOPMENT AND TESTING

R version 3.4.1 (2017-06-30)
abind : version 1.4.5
reshape2 : version 1.4.3
ggplot2 : version 3.1.0
magrittr : version 1.5
dplyr : version 0.7.4
Matrix : version 1.2.10
xlsx : version 0.6.1
stats : version 3.4.1
graphics : version 3.4.1
grDevices : version 3.4.1
utils : version 3.4.1
datasets : version 3.4.1
methods : version 3.4.1
base : version 3.4.1