SIMULATING PTA-STYLE DATA SETS

DOCUMENTATION

Version 1.3

06 Dec 2014

G. Hobbs

Some of the routines built into ptaSimulate are based on TOASIM code developed initially by M. Keith.

Help in developing this software has been provided by (in no particular order):
W. Coles, S. Dai, R. Shannon, M. Kerr, C. Tiburzi, D. Reardon

Users/testers: R. van Haasteren, J. Ellis, S. Taylor

DOCUMENTATION VERSION HISTORY

Version #	Implemented By	Revision Date	Reason
1.0	*G. Hobbs*	*10 Oct 2014*	*Creation of document*
1.1	G. Hobbs	12 Nov 2014	Major revision now that the structure of the code is better defined.
1.2	G. Hobbs	28 Nov 2014	• Simple instrumental errors • Individual GW sources
1.3	G. Hobbs	06 Dec 2014	Added in simulation of outliers and tidied up the code a bit.

TABLE OF CONTENTS

1 OVERVIEW

The ptaSimulate software has been developed to simulate output data files from a pulsar timing array (PTA) project. The majority of the code is based on simulating realistic pulse arrival times, but the code can also produce simulated pulse profiles. The inspiration for the project came from:

- *the difficulty that people had in using the simulation software within tempo2 for their research.* Numerous papers have been based on tempo2 simulations. These include searches for gravitational wave memory events or individual sources, placing bounds on the gravitational wave background amplitude, studies of the possibility of false detections and ensuring that pulsar parameters measured using the Cholesky methods are not biased.

- *the difficulty in quickly creating tempo2 simulations*. Typical users of the tempo2 simulation software make repeated calls to tempo2 and are unable to parallalise their code to run on multiple nodes or on a supercomputer.

- *queries on "when will we detect gravitational waves"?* This is a fundamental question about the PTA projets and recent predictions have disagreed with each other.

- *queries on "how best can the PPTA project fit into the IPTA"?* The PPTA team is considering changing observing strategy to improve the sensitivity of the IPTA data sets. However, we currently have no methods to study this problem.

- *queries on "how much worse would the PPTA data sets be if we replaced the multibeam receiver with a PAF"?* CSIRO is planning on installing a PAF at Parkes for many months. This PAF will have significantly higher Tsys than the existing multibeam receiver and therefore the PPTA data will be degraded.

Note that the ptaSimulate software does not attempt to answer these questions directly. Instead it provides an easy way to simulate tempo2 parameter (.par) and arrival time (.tim) files. Those files can be run through other pipelines (such as a GWB detection pipeline) as if they were actual observations.

2 INTRODUCTION

The software uses an input text file that must be created by the user. This input file can have any filename and provides a description of the observing system to be simulated. The software is run as:

- ./ptaSimulate <filename>

The software automatically produces a set of directories in the local directory. It then partially fills those directories with output files and also produces a set of tcsh scripts that need to be run to complete the simulations. Typical usage would be:

- ./ptaSimulate <filename>; source <name>/scripts/runScript_master

The output is placed into <name>/output/<real_X>/ where X is realization number. That directory contains .par and .tim files for each simulated pulsar.

The following provides a summary of what can be simulated by ptaSimulate (those in a red font have not been implemented yet):

- Realistic pulsar parameters
- Realistic pulsar positions
- Time between sessions
- Number of observations during observing sessions
- Timing noise
- Timing noise defined using beta
- Jitter noise
- Kolmogorov DM variability
- DM variability defined by a covariance function
- dDM given by arbitrary functional form
- ToA uncertainties from the radiometer equation and pulse shape
- Diffractive scintillation
- Isotropic, stochastic gravitational wave background
- Instrumental jumps
- Instrumental errors and artefacts
- Outliers
- Glitches and glitch recovery
- The solar wind
- Gravitational wave burst events (A+ and Ax)
- Gravitational wave memory events
- Continuous wave sources
- Real errors in the terrestrial time standard

- Red noise errors in the terrestrial time standard (defined, if required, by the Allan variance)
- Real errors in planetary ephemerides
- Errors in the Earth-orientation files
- Realistic rise and set times
- Red noise errors in the planetary ephemeris
- Right-circular errors in the planetary ephemeris
- Timing noise induced by a two-state process
- Detailed scintillation model
- GWM pulsar term
- Refraction
-
- Polarisation calibration errors
- Realistic observation times during a session
- RFI
- Non-isotropic gravitational wave background
- Atmospheric effects
- Scattering delays
-
- Simulating moving observatory (i.e., navigation)
- Add anything into the sim parameter file
- Add anything into a global file

Still to do internally

- Ability to hold some parameter fixed
- Use of PSRPOP output
- Setting pulsar distances
- Update plotMany to be able to lock the y-scales for different pulsars.
- Simulating subints and subchannels
- Making psrfits output profiles
- Allowing user selection between NONE, PSRFITS and ASCII profiles for output
- Lots of checking that parameters make sense

The code is relatively straightforward to compile as it is completely written in C and only uses the fftw external library. Compilation will be similar to:

gcc -lm -o ptaSimulate *.c -lfftw3 /usr/lib/libfftw3f.so

For general usage of ptaSimulate, the user also requires that the following programs are installed:

- psrcat (with up-to-date database file) – see http://www.atnf.csiro.au/research/psrcat
- tempo2 and $TEMPO2 is set.

In order to make profile templates to simulate realistic error bar sizes then the following software is required:

- ptime – this is used for making analytic templates with the correct format. Give download and installation instructions.

3 EXAMPLE USAGE

3.1 SIMPLE SINGLE PULSAR SIMULATION

The following script demonstrates how to make a very simple simulation. A single pulsar (J0437-4715) is simulated. The data sampling is regular (20 day cadence) and all ToAs have the same frequency (1400 MHz) and uncertainty (1us).

```
<define>
name: sim1
nproc: 2
nreal: 1
</define>
```

This section defines the properties of the simulation. The "name" will be used as the base directory name that will contain the results and intermediate files. Nproc is the number of processors available and nreal is the number of realisations

```
<pulsars>
psr: name=J0437-4715
</pulsars>
```

If only pulsar names are given then their parameters are automatically obtained from running "psrcat" on the command line.

```
<obsRun>
name: pks
tel: pks
start: 53371
finish: 56000
sampling: cadence=20
sched: sched1
</obsRun>
```

This defines the telescope at which the data will be simulated (standard tempo2 telescope code names, e.g., pks for the Parkes telescope can be used), the start and end times of the observation run, the observation sampling (cadence is given in days) and which schedule is run during each observing session. Note that multiple <obsRun> definitions can be used.

```
<schedule>
name: sched1
observe: psr=J0437-4715,toaerr=1e-6,freq=1400
</schedule>
```

A simple definition of a single observation of PSR J0437-4715 with a ToA uncertainty of 1us and frequency of 1400 MHz during each observing session.

This gives the figure shown below:

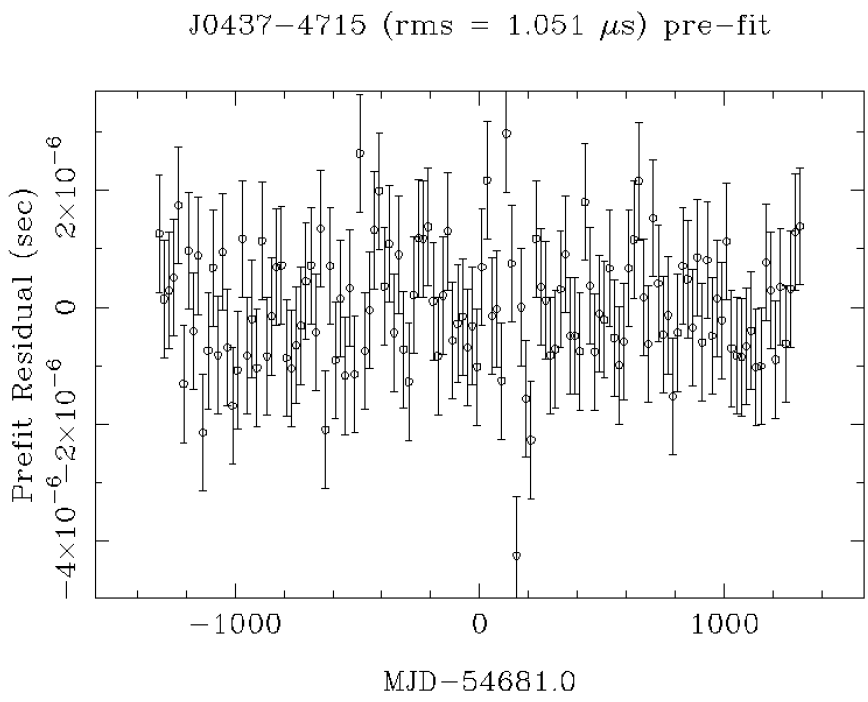

Figure 1: Simple simulation of a single pulsar dataset affected by white noise

3.2 MULTIPLE PULSARS WITH DIFFERENT START TIMES AND RMS RESIDUALS

As the next example shows, it is easy to simulate multiple pulsars.

```
<define>
name: sim2
nproc: 2
nreal: 1
</define>

<pulsars>
psr: name=J0437-4715
psr: name=J1022+1001
psr: name=J1909-3744
</pulsars>

<obsRun>
name: pks
tel: pks
start: 53371
finish: 56000
sampling: cadence=20
sched: sched1
```

Here we provide all three pulsars that we wish to simulate.

```
</obsRun>

<schedule>
name: sched1
observe: psr=J0437-4715,toaerr=1e-7,freq=1400
observe: psr=J1022+1001,toaerr=1e-6,freq=1400,start=55000
observe: psr=J1909-3744,toaerr=3e-7,freq=1400,start=54000,finish=56000
</schedule>
```

We require details of the observations for each pulsar.

Using plotMany to show the output gives the following plot:

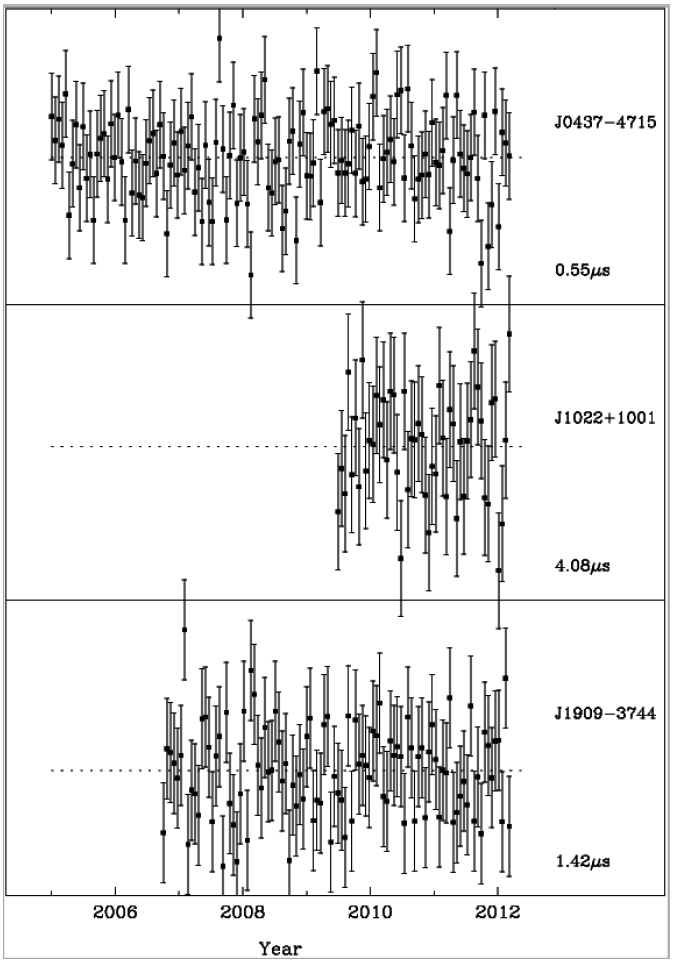

Figure 2: Simulation of three pulsars simultaneously all in the 20cm observing band. This plot was made using tempo2-dev -gr plotMany -f J0437-4715.par J0437-4715.tim -f J1022+1001.par J1022+1001.tim -f J1909-3744.par J1909-3744.tim -reverse -centremjd -1 -fontsize 0.9 -plotus

3.3 REALISTIC RISE AND SET TIMES

The simulation described above has a simple, constant observing cadence. However, it is clearly not possible to observe a pulsar at a time when the source has set. It is also common to observe pulsars at a similar LST during each observing run. Such information can be defined as:

```
<schedule>
name: sched1
observe: psr=J0437-4715,toaerr=1e-6,freq=1400, ha=8*(ran(linear)-0.5)
observe: psr=J1022+1001,toaerr=1e-6,freq=1400,ha=2
observe: psr=J1909-3744,toaerr=1e-6,freq=1400
</schedule>
```

In the first case (J0437-4715) the hour angle is chosen randomly between +/- 4 hours. In the second case (J1022+011) a particular hour angle is chosen (2 hours) and in the final case (J1909-3744) no hour angle limits are provided.

Determining the transit time for a pulsar at a given observatory requires knowledge of the observatory coordinates. This is obtained from the $TEMPO2/observatory/observatories.dat file. It is therefore essential that the observatory name is the same as that defined in the tempo2 file.

3.4 MULTIPLE PULSARS WITH MULTIPLE OBSERVING FREQUENCIES

In the examples above we assumed that all observations were carried out at 1400 MHz. Simulating different observing bands is straightforward. The observing frequencies can be specified for each simulated observation.

```
<define>
name: sim3
nproc: 2
nreal: 1
</define>

<pulsars>
psr: name=J0437-4715
psr: name=J1022+1001
psr: name=J1909-3744
</pulsars>

<obsRun>
name: pks
tel: pks
start: 53371
```

```
finish: 56000
sampling: cadence=20
sched: sched1
</obsRun>

<schedule>
name: sched1
observe: psr=J0437-4715,toaerr=1e-7,freq=1400
observe: psr=J0437-4715,toaerr=5e-7,freq=730,start=54000
observe: psr=J0437-4715,toaerr=2e-6,freq=3100,start=54000

observe: psr=J1022+1001,toaerr=1e-6,freq=1400,start=55000
observe: psr=J1022+1001,toaerr=0.8e-6,freq=730,start=55000
observe: psr=J1022+1001,toaerr=1.5e-6,freq=3100,start=55000

observe: psr=J1909-3744,toaerr=3e-7,freq=1400,start=54000,finish=56000
observe: psr=J1909-3744,toaerr=3e-7,freq=730,start=54000,finish=56000
observe: psr=J1909-3744,toaerr=1e-7,freq=3100,start=54000,finish=56000
</schedule>
```

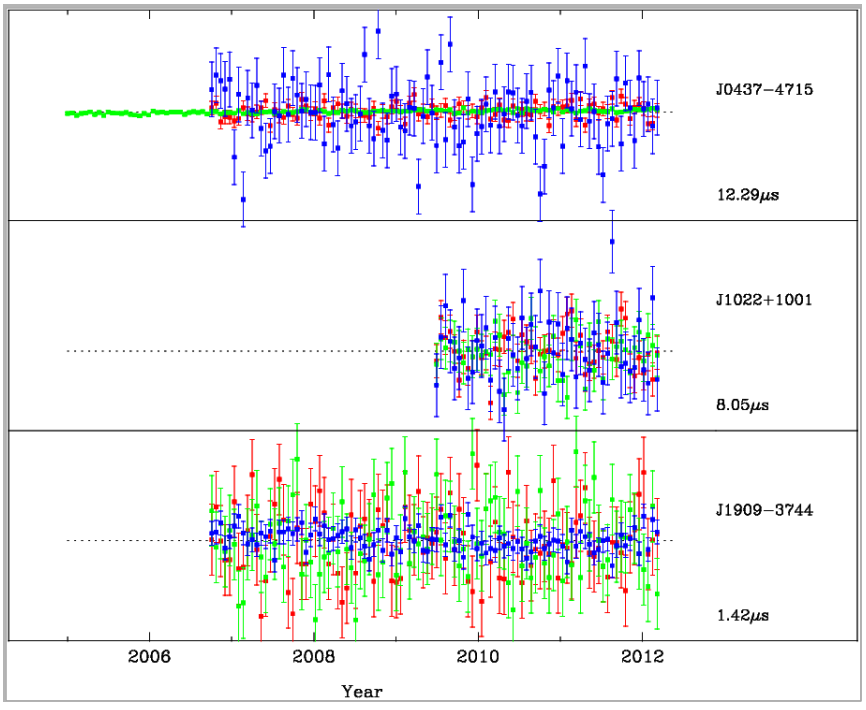

Figure 3: Simulation of three pulsars observed with three observing bands (10, 20 and 50cm)

3.5 ADDING IN DISPERSION MEASURE VARIABILITY

The above simulations assume that the dispersion measure (DM) is constant. There are currently three methods for changing the DM as a function of time. The first is to assume

a Kolmogorov spectrum, the second is to add in a model for the dDM covariance function and the third is to provide an analytic expression for the DM variability. These three methods are demonstrated in the example below for J0437-4715, J1022+1001 and J1909-3744 respectively (note that the simulation is just for example and does not try and reproduce realistic DM variability in these pulsars).

```
<define>
name: sim4
nproc: 2
nreal: 1
</define>

<pulsars>
psr: name=J0437-4715
psr: name=J1022+1001
psr: name=J1909-3744
</pulsars>

<obsRun>
name: pks
tel: pks
start: 53371
finish: 56000
sampling: cadence=20
sched: sched1
</obsRun>

<add>
dmvar: psr=J0437-4715,D=[1000;1400;30]
dmCovar: psr=J1022+1001,alpha=2,a=1.3e-7,b=331
dmFunc: psr=J1909-3744,ddm=(x-55000)*1e-6+1e-3*exp(-(x-54500)**2/(2*100*100))
</add>

<schedule>
name: sched1
observe: psr=J0437-4715,toaerr=1e-7,freq=1400
observe: psr=J0437-4715,toaerr=5e-7,freq=730,start=54000
observe: psr=J0437-4715,toaerr=2e-6,freq=3100,start=54000

observe: psr=J1022+1001,toaerr=1e-6,freq=1400,start=55000
observe: psr=J1022+1001,toaerr=0.8e-6,freq=730,start=55000
observe: psr=J1022+1001,toaerr=1.5e-6,freq=3100,start=55000

observe: psr=J1909-3744,toaerr=3e-7,freq=1400,start=54000,finish=56000
observe: psr=J1909-3744,toaerr=3e-7,freq=730,start=54000,finish=56000
```

observe: psr=J1909-3744,toaerr=1e-7,freq=3100,start=54000,finish=56000
</schedule>

Different parameters defined on the same line in the input file (such as alpha=2,a=1.3e-7,b=331) are separated by commas. Single parameters that take multiple values (e.g., D=[1000;1400;0.3] are defined using square brackets and then separated using semi-colons.

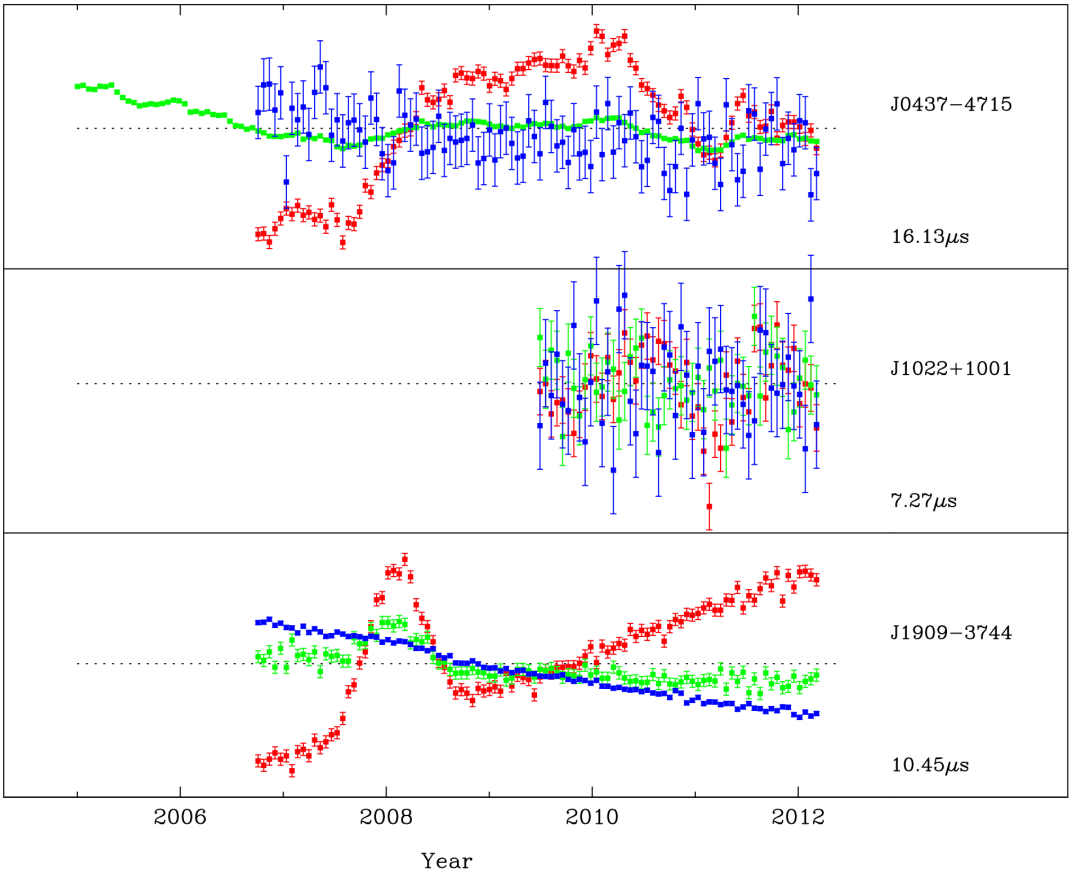

Figure 4: Three different methods to simulate dispersion measure variations

3.6 ADDING TIMING NOISE

Many phenomena induce timing residuals that exhibit red spectra.

Two definitions exist for the red noise power spectral density:

$$P_1 = \frac{P_0}{\left(1 + \left[\frac{f}{f_c}\right]^2\right)^{\frac{\alpha}{2}}}$$

and for timing noise that exhibits low-Q oscillations

$$P_2 = \frac{P_0 \left(\frac{f}{f_c}\right)^{\beta}}{\left(1 + \left[\frac{f}{f_c}\right]^2\right)^{\frac{\alpha}{2}}}$$

The timing noise can also be specified through a model of its covariance function.

<add>
tnoise: psr=J0437-4715,alpha=-4,p0=1e-24,fc=0.2
tnoise: psr=J0613-0200,alpha=-6,beta=3,p0=1e-27,fc=0.4
</add>

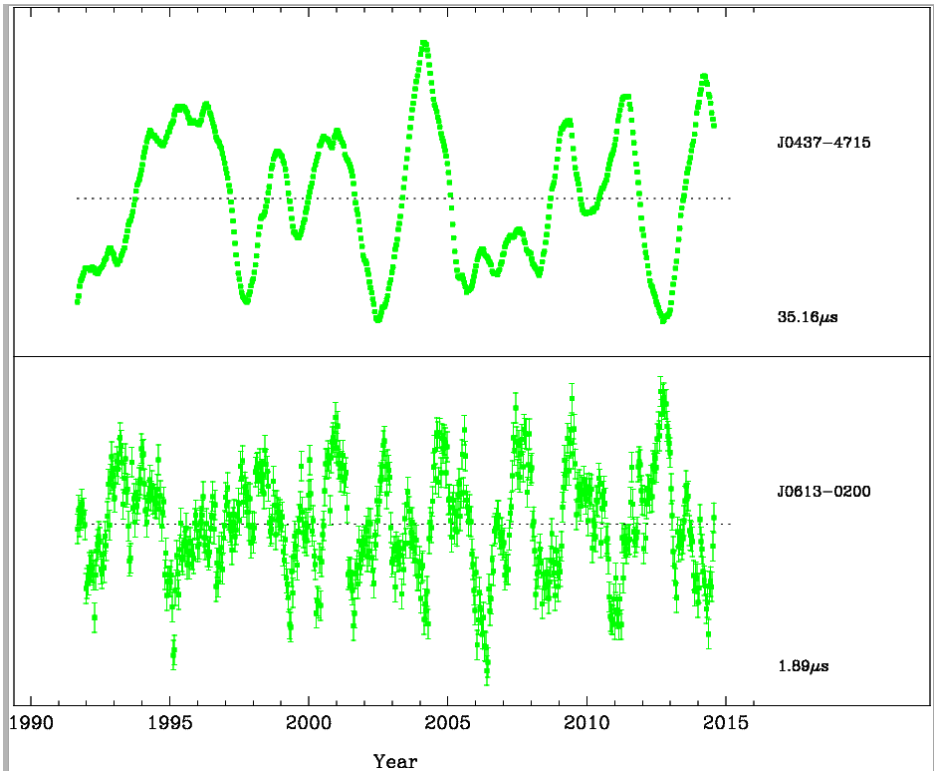

Figure 5: Demonstration of two methods for simulating timing noise

For timing noise that is correlated between pulsars see the sections below on simulations gravitational waves, clocks and ephemeris errors.

It is also possible to add in uncorrelated red noise that has the same spectral properties as a gravitational wave background with a given amplitude:

<add>
tnoise: psr=J0437-4715,gwamp=1e-15

</add>

3.7 ADDING GLITCHES

Glitch events can easily be added. The following adds two glitches into the residuals for J0437-4715.

<add>
glitch: psr=J0437-4715, glep=54000,glph=0,glf0=1e-11,glf1=0
glitch: psr=J0437-4715, glep=55000,glph=0,glf0=1e-11,glf1=0
</add>

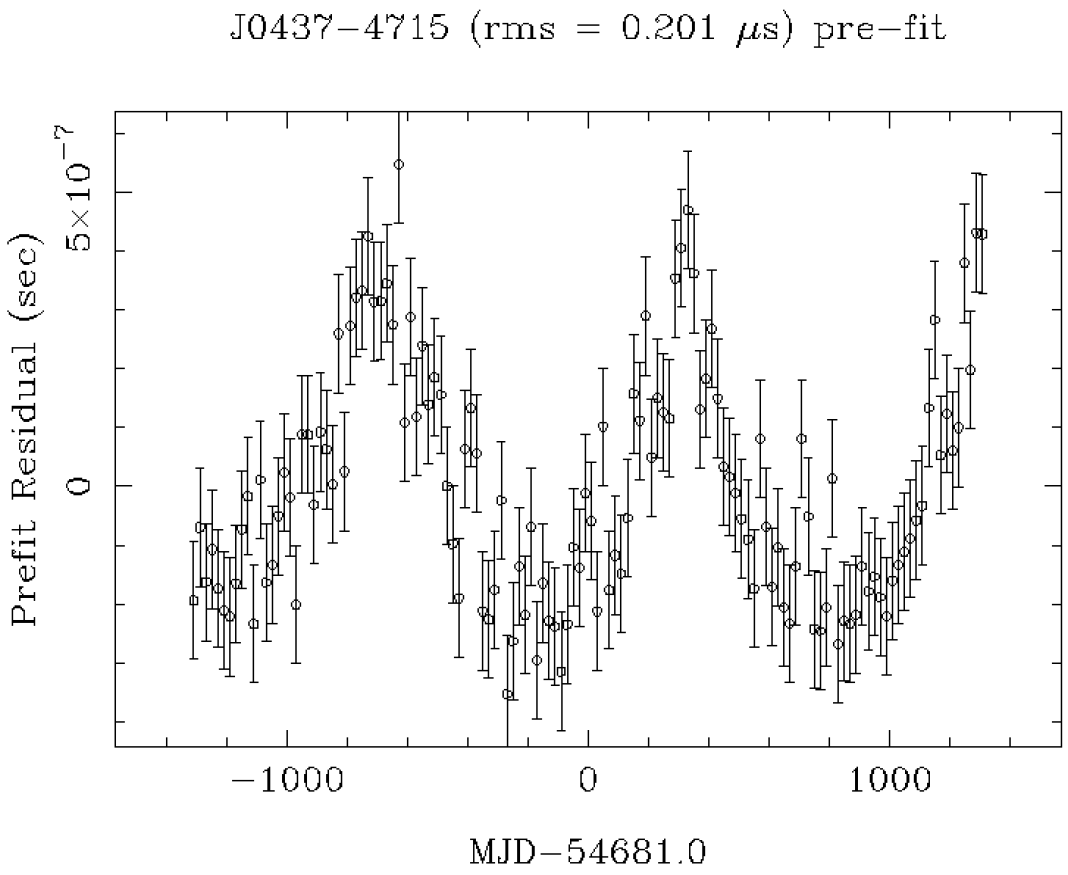

Figure 6: Two glitch events (at MJD 54000 and 55000)

The following parameters can be used:

- Glep – epoch of glitch
- Glph – phase change at glitch
- Glf0 – frequency change at glitch

- Glf1 – frequency derivative change at glitch
- Glf0d – pulse frequency increment
- Gltd – decay time constant

Note that some timing noise can be modeled as sudden switches between two spin-down rates. This can be modeled by simulating multiple glitch events in which only the GLF1 is set.

3.8 ADDING JITTER NOISE

Jitter noise is defined as follows:

<add>
jitter: psr=J2145-0750,SJ=[3600;3100;420e-9]
</add>

where SJ (σ_J) defines the rms residual induced by the jitter at the given timescale (3600s in the above example) and frequency (3100MHz). The value is the third argument. The amount of jitter is subsequently calculed as:

$$\sigma = \sigma_J \sqrt{\frac{t_{ref}}{t_{obs}}}$$

If the jitter parameters are unknown then they can be modeled based on the pulsar's properties (PROBABLY REQUIRE DEFINITION OF PULSE WIDTH – IF WIDTH DEFINED THEN ALLOW SIMPLE CALCULATION OF TOA ERR BASED ON WIDTH).

DISCUSS BROADBAND NATURE OF JITTER AND OBSERVING TIMES

3.9 ADDING GRAVITATIONAL WAVES

3.9.1 A stochastic, isotropic background

Currently only a stochastic and isotropic gravitational wave background (GWB) can be simulated. The only option is the amplitude of the GWB given by:

$$h = A\left(\frac{f}{f_{yr}}\right)^{\alpha}$$

An example of defining the GW parameters:

<add>
gwb: amp=1e-15
</add>

which will simulate a GWB background in the timing residuals for all pulsars with default exponent (alpha = -2/3) and an amplitude of 1e-15. Simulating gravitational waves requires knowledge of the pulsar distances. If the pulsar parameters are obtained from the PSRCAT database then the distance is determined from the best available distance (DIST) check that this is correct. For simulated pulsars then the distance is determined as:

$$dist(pc) = \frac{DM\ (cm^{-3}pc)}{0.03(cm^{-3})}$$

or the distance can be directly specified when defining the pulsar using dist=XX.

<add>
gwb: amp=1e-15
</add>

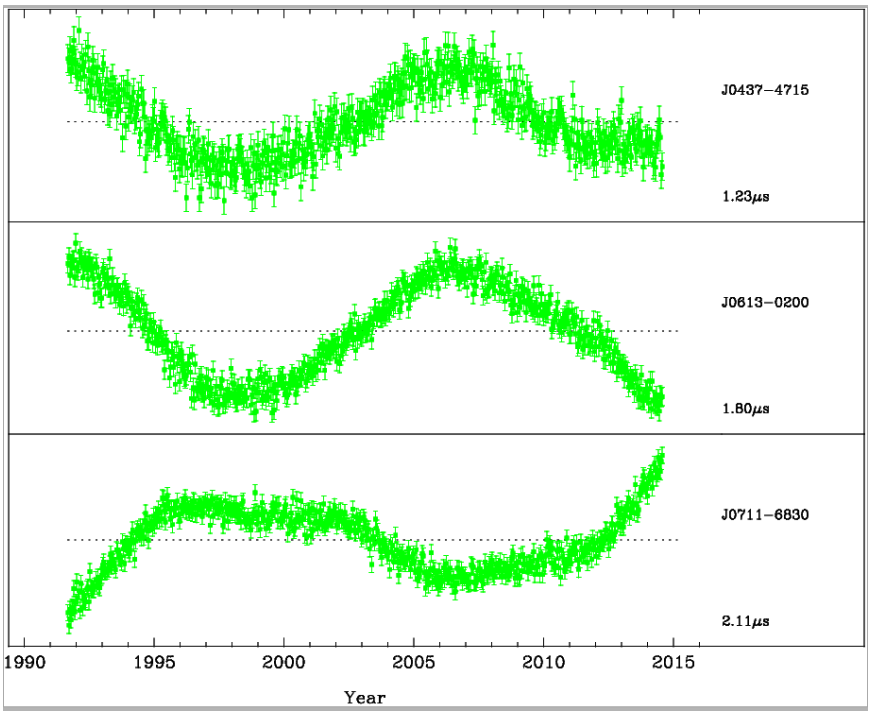

Figure 7: Addition of gravitational wave background (A=10-15) to three pulsar datasets

3.9.2 A general background formed from single sources

- Have Vikram's implementation and ensure that we can use Alberto's files

3.9.3 A single supermassive binary black hole

The effects of a single supermassive binary black hole system can be simulated using the formalism of Zhu et al. The required input parameters are the GW source position, frequency (Hz), H0, epoch of definition, cosine of the inclination angle, polarization angle (radians) and the chirp mass (solar masses). If the chirp mass is set to zero then the pulsar term is simulated at the same frequency as the Earth term.

```
<add>
gwsingle: ra=0.3,dec=-0.3, cgw_freq=8e-8,
   cgw_h0=2e-13,cgw_epoch=52000,cgw_cosinc=1,cgw_angpol=0,cgw_mc=1e9
</add>
```

For five simulated pulsars this gives the residuals in the left-hand panel of the figure below. Both the Earth term (high frequency signal) and pulsar term (low frequency signal). The right-hand panel shows the post-fit residuals after fitting for the Earth term using the A+ and Ax formalism within tempo2. The pulsar-term signal remains, but the Earth-term signal has been removed.

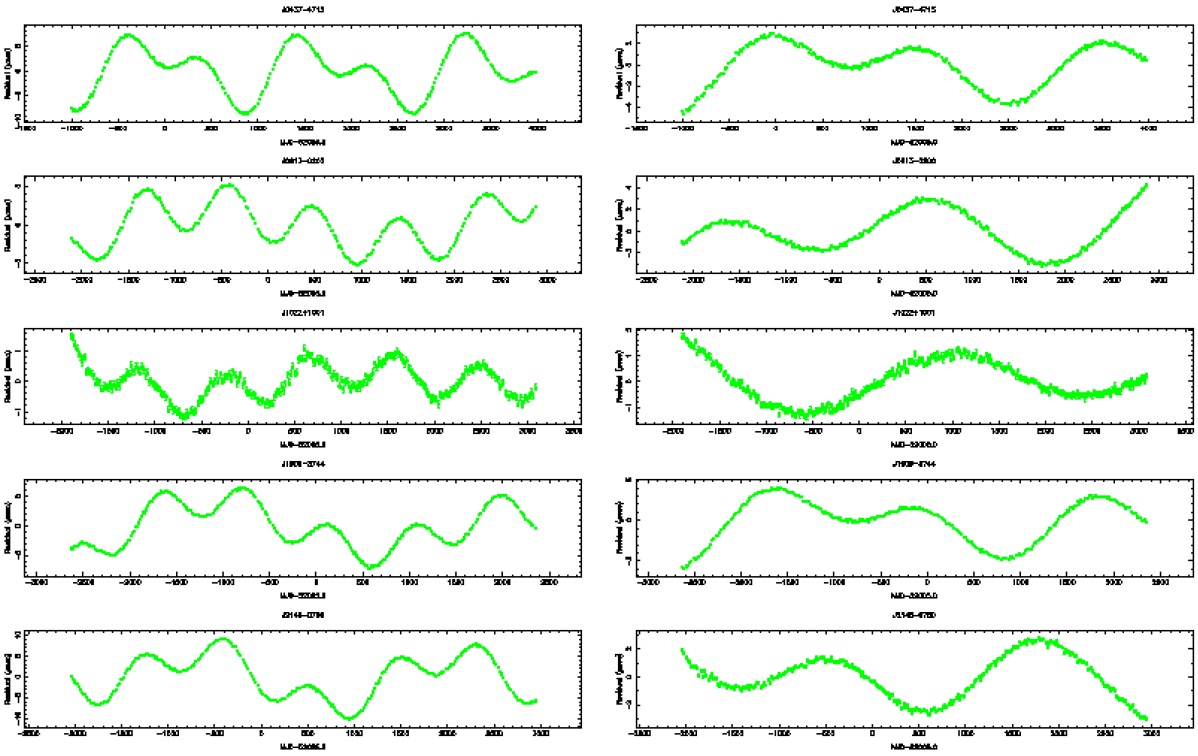

Figure 8: Simulating timing residuals containing a slowly evolving CW source.

Adding in multiple single sources is straightforward:

```
<add>
gwsingle: ra=0.3,dec=-0.3, cgw_freq=8e-8,
   cgw_h0=2e-13,cgw_epoch=52000,cgw_cosinc=1,cgw_angpol=0,cgw_mc=1e9
gwsingle: ra=0.6,dec=0.3, cgw_freq=3e-8,
   cgw_h0=3e-13,cgw_epoch=52000,cgw_cosinc=1,cgw_angpol=0,cgw_mc=1e9
</add>
```

To switch off the pulsar term use pulsarTerm=0.

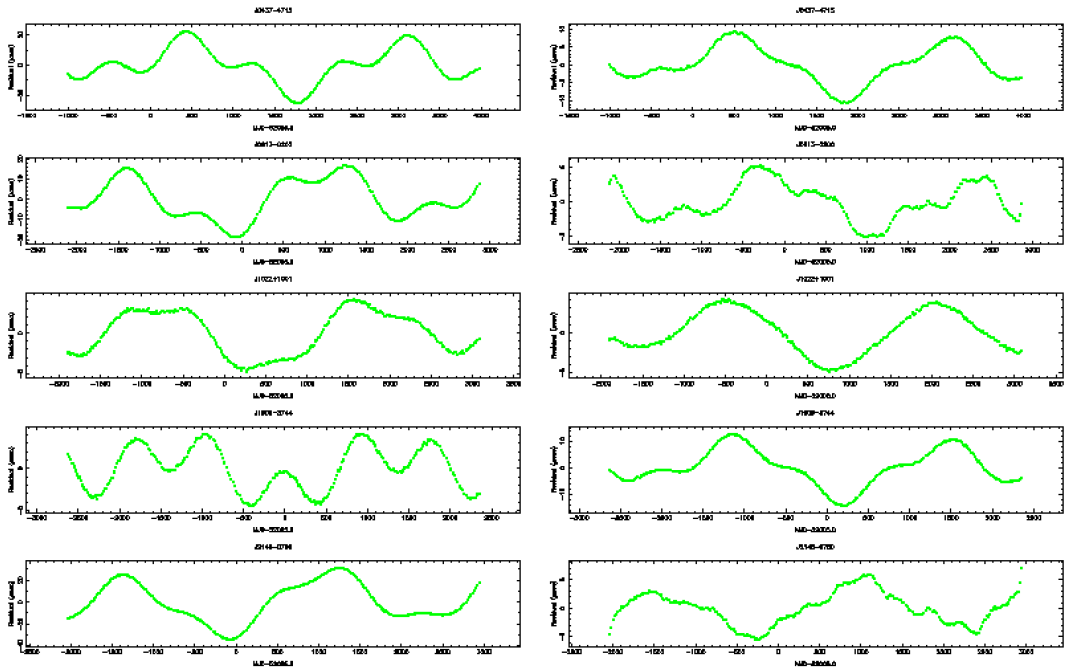

Figure 9: Simulation of two single sources where the chirp mass = 0

3.9.4 Gravitational wave memory

Wang et al. (2014) implemented a model for the gravitational wave memory (GWM) effect into tempo2. His implementation can be used as follows:

<add>
gwsingle: ra=0.2,dec=-0.3,gwm_amp=1e-12,gwm_epoch=54500,gwm_phi=0.3
</add>

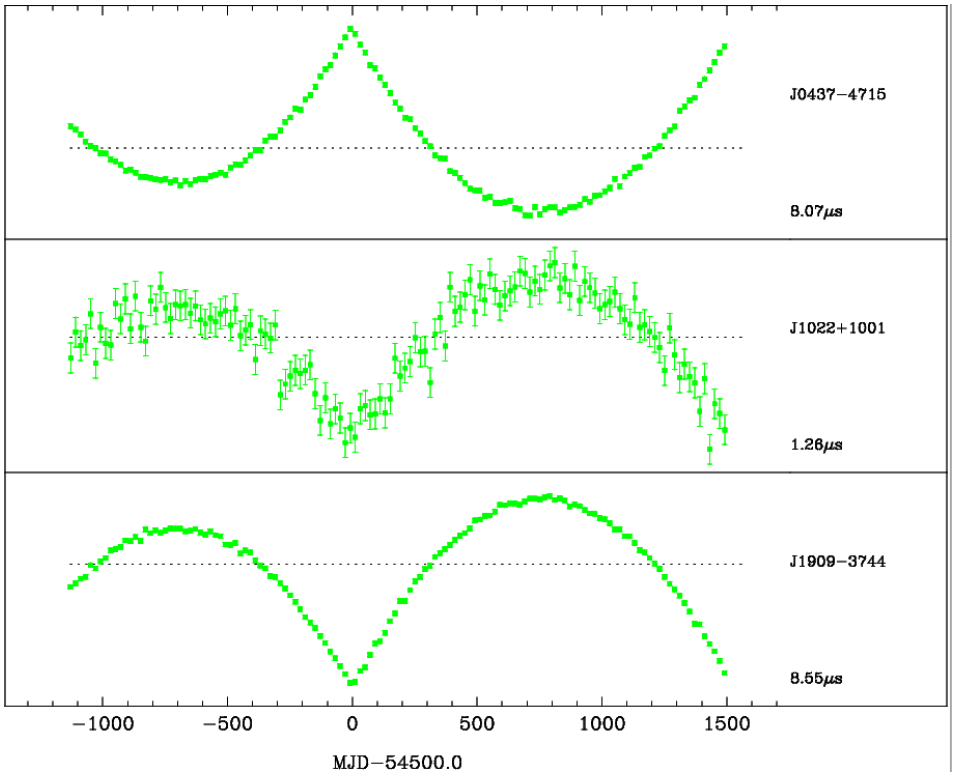

Figure 10: Simulation of the gravitational wave memory effect

3.9.5 Gravitational waves of arbitrary functional form

It is possible to define analytic expressions for the two polarization states of the GW signal: A+ and Ax as follows:

<add>
gwsingle: ra=0.2,dec=-0.3,ap=1e-5*exp(-(x-53000)**2/1e5)),ac=-0.8e-5*exp(-(x-53100)**2/3e5))
</add>

This gives the output as shown in the figure below for three pulsars. Note that the A+ and Ax code within tempo2 is not used for this simulation. The expressions are evaluated at each ToA.

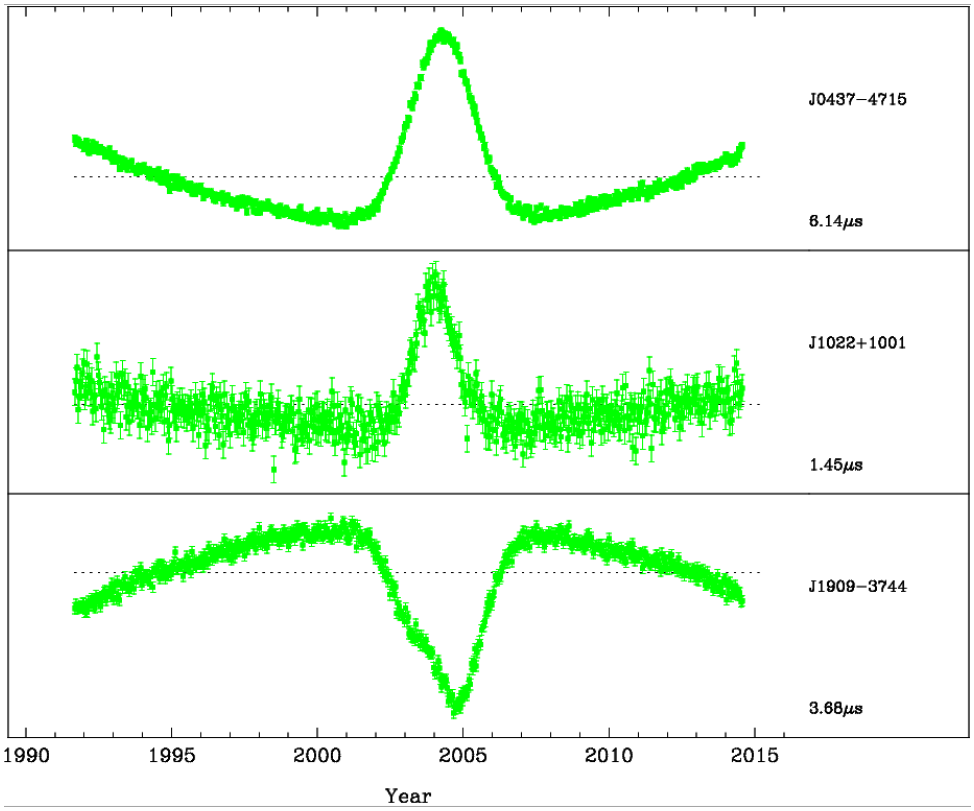

Figure 11: Simulation of a Gaussian gravitational-wave burst event

A GW memory event can be simulated using condition expression operations such as:

<add>
gwsingle: ra=0.2,dec=-0.3,ap=(x > 53000) ? 0 : 1e-8*(x - 53000),ac=0
</add>

This gives the following figure:

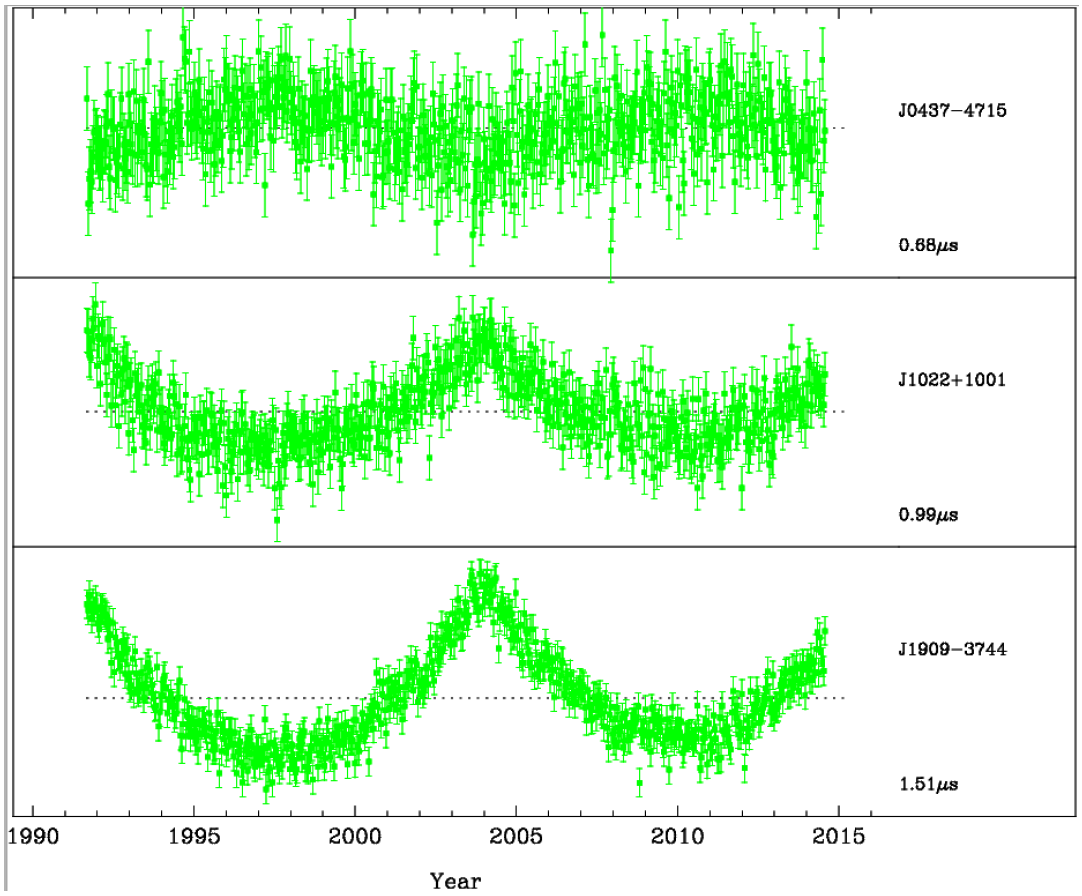

Figure 12: Simulation of a gravitational wave memory event simulated using a definition of A+ and Ax

3.10 CLOCK NOISE

- Allan variance
- Observatory clock noise

There are multiple methods for simulating clock noise. The following methods are used to produce a signal that is common to all pulsars. This will be true for noise in the reference time standard (such as TT(TAI)) or for all pulsars at the same observatory.

The first method is to simulate red noise, but ensure that the same red noise is simulated for all pulsars. This can be carried out using:

<add>
clknoise: alpha=-4,p0=1e-26,fc=0.02
</add>

where the parameter definitions are the same as for the timing noise simulations.

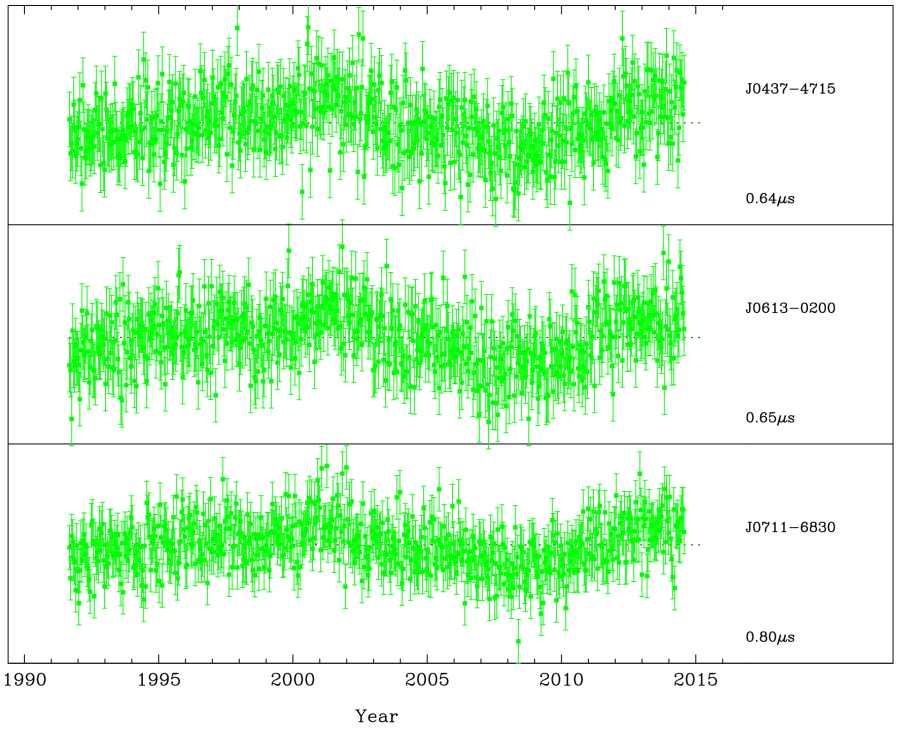

Figure 13: Simulation of clock noise defined from a simple power-law red noise model

It is also possible to use the actual difference between two different time standards. The following example simulates the ideal site arrival times using the best available time standard, TT(BIPM), but processes the data using International Atomic Time (TAI).

<t2files>
clk: use=TT(TAI),sim=TT(BIPM2013)
</t2files>

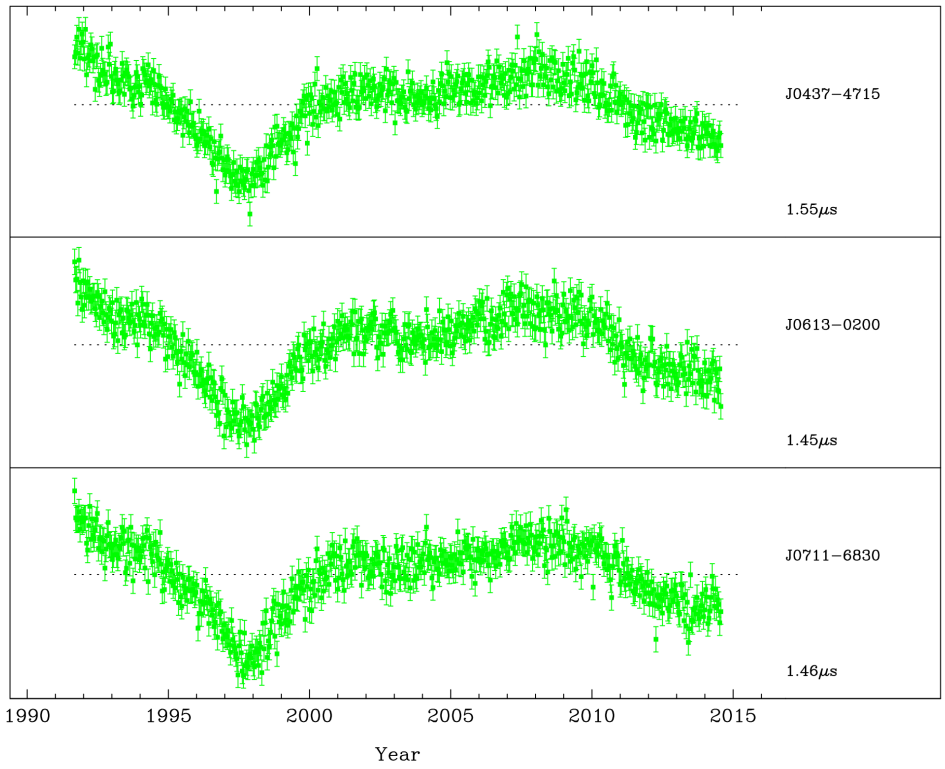

Figure 14: Residuals affected by the difference between TT(BIPM) and TT(TAI)

3.11 EPHEMERIS ERRORS

- Adding red ephemeris noise
- Adding right-hand circular ephemeris noise

Errors in the solar system ephemeris can lead to correlated residuals. In the example below the JPL DE421 solar system ephemeris is used to create the idealized site arrival times. The DE414 ephemeris (i.e., an earlier ephemeris) is used to process the data.

<t2files>
ephem: use=DE414,sim=DE421
</t2files>

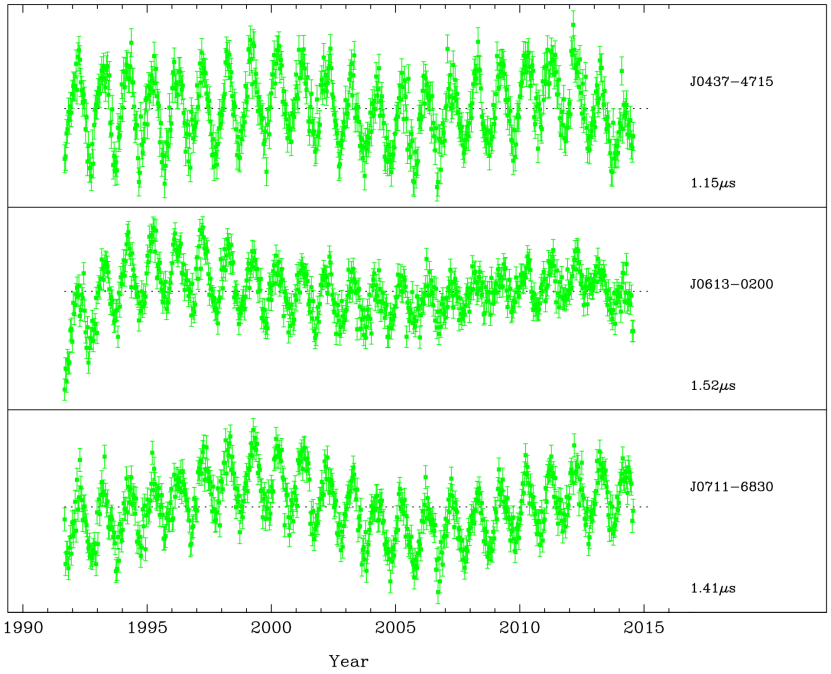

Figure 15: Residuals showing differences between two solar system ephemerides

In this case the 1-year periodicity is caused by a change in the reference frame between the two ephemerides. This would be removed by fitting for each pulsar's position and proper motion.

3.12 EARTH ORIENTATION PARAMETERS

The Earth orientation parameter file can be specified using the eop definition. In the following example the EOPC file has been truncated in the year 2005. Note that the EOP file must be in $TEMPO2/.

<t2files>
eop: sim=earth/eopc04_stop2005
</t2files>

This gives the following residuals.

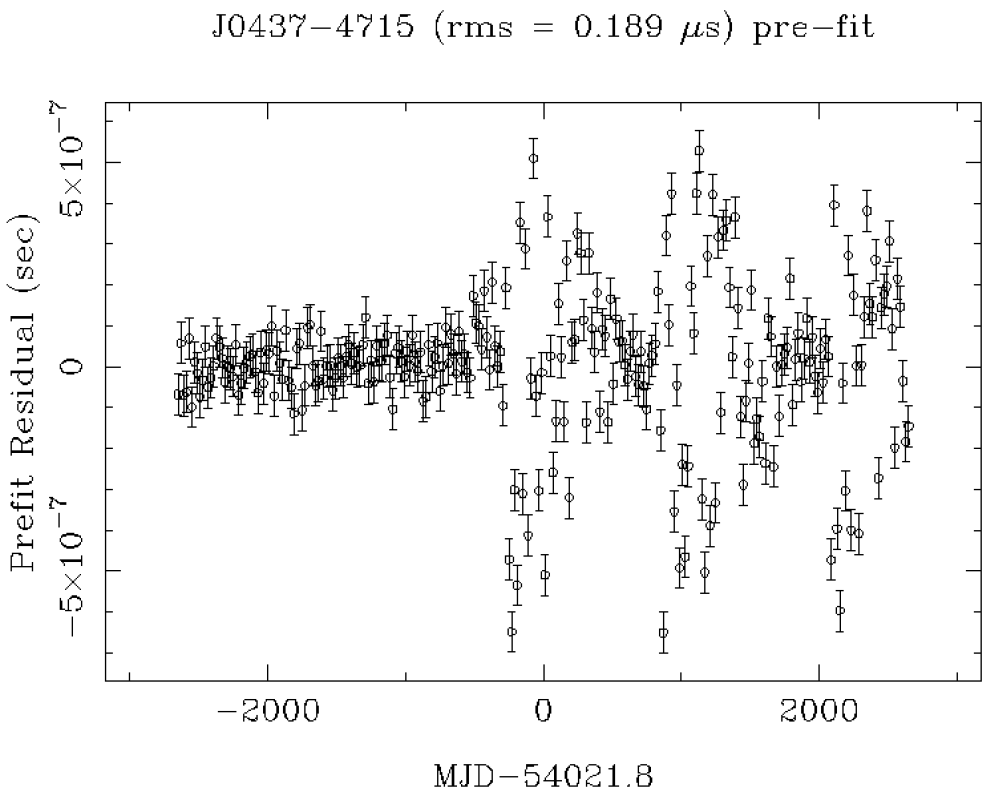

Figure 16: Resulting residuals after using an incorrect EOP input file

3.13 THE SOLAR WIND

The default solar wind model in tempo2 is very simplistic. You et al. showed how more realistic solar wind models can be constructed using actual observations of the Sun by the Wilcox Solar Observatory (WSO). These data files can be used in ptaSimulate as:

<t2files>
swm: sim=wso
</t2files>

which defines for the simulations (sim=) or for the final parameter file (use=) that data files from the WSO should be used. For PSR J1022+1001 this leads to the simulated residuals as shown below.

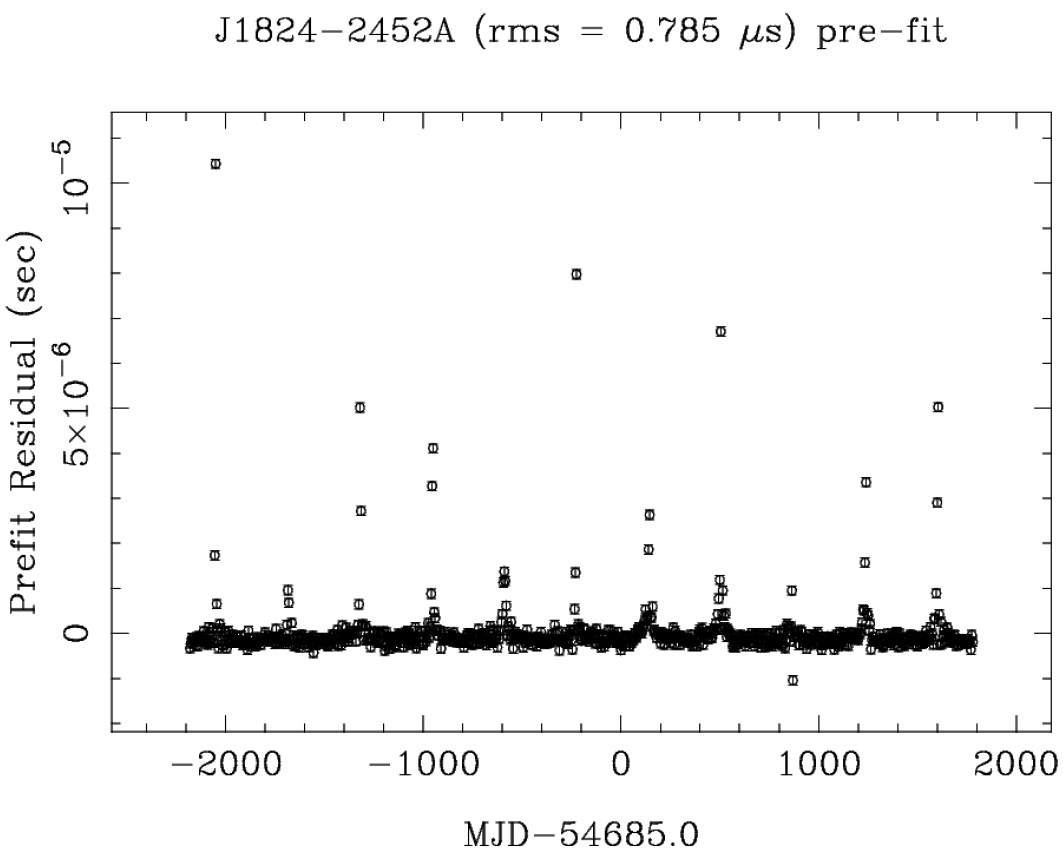

Figure 17: The effect of the solar wind on the timing residuals for PSR J1824-2452A

3.14 OBSERVING SESSIONS

At most observatories the project team will request a certain amount of observing time. That time is usually provided in blocks, i.e., a few days every few weeks. However, it is not usually possible to organize a non-changing observing cadence. Also, instrument failures, wind, etc. can lead to observations being missed.

ptaSimulate allows the user to:

- Attempt realistic simulations of the time between sessions
- To make use of actual observation times (if available)

3.14.1 Simulating time between observing sessions

The following simulates observations exactly every 20 days:

<obsRun>
name: test
tel: pks

start: 49000
finish: 49421
sched: sched1
sampling: cadence=20
</obsRun>

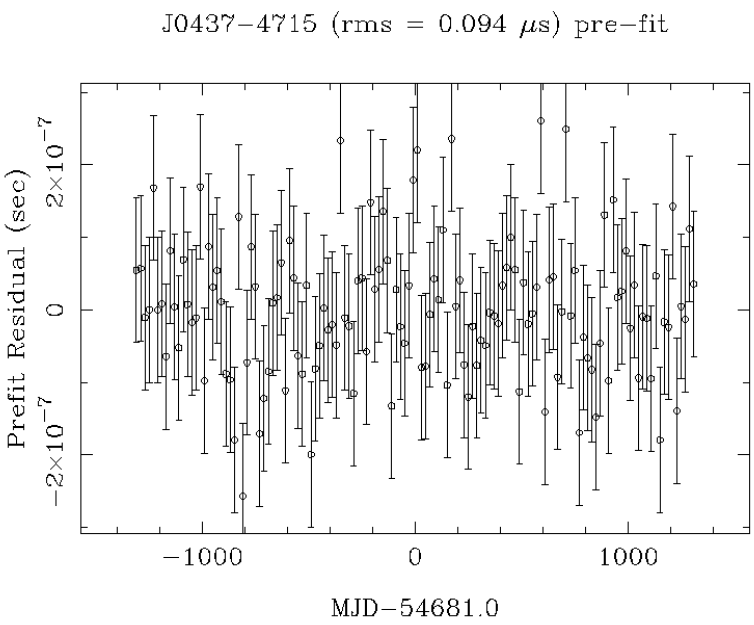

Figure 18: Simulation of observations with a regular 20 day cadence

It is possible to add some randomness into the sampling:

<obsRun>
name: test
tel: pks
start: 49000
finish: 49421
sched: sched1
sampling: cadence=100+100*ran(linear)
</obsRun>

This will randomly select the time between observing sessions to be between 0 and 200 days.

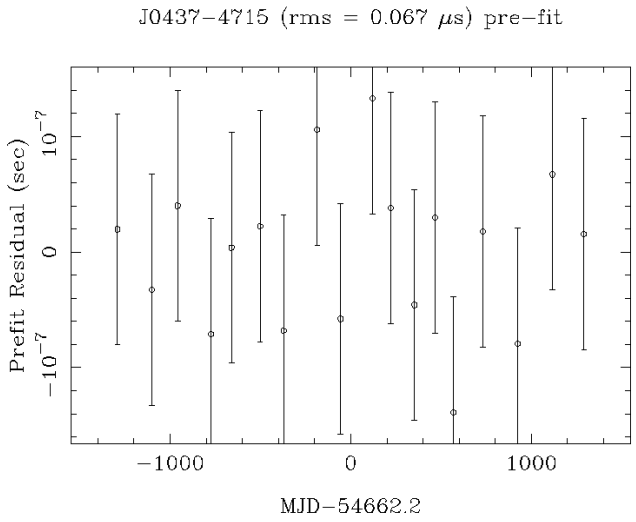

Figure 19: Simulation in which the observations are not regularly spaced

The probability of an entire observing session being missed can be defined as:

<obsRun>
name: test
tel: pks
start: 49000
finish: 49421
sched: sched1
sampling: cadence=15+10*ran(linear), probFailure=ran(linear)>0.5
</obsRun>

In this case the probability of failure is 1 in 2. Of course, other distributions (such as a Gaussian distribution) can be used.

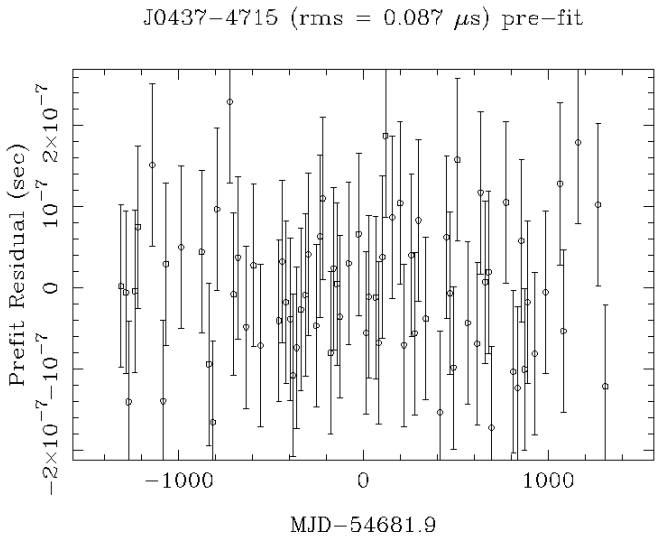

Figure 20: Simulation in which some observing sessions have been missed

3.14.2 Realistic observations in a session

- Number of observations
- Probability of failure
- Rise and set times

3.14.3 Using actual observations

<define>
name: sim7
nproc: 1
nreal: 1
t2exe: tempo2-dev
</define>

<pulsars>
psr: name=J1713+0747
</pulsars>

<obsRun>
name: pks
tel: pks
t2tim: psr=J1713+0747,file=J1713+0747.tim
</obsRun>

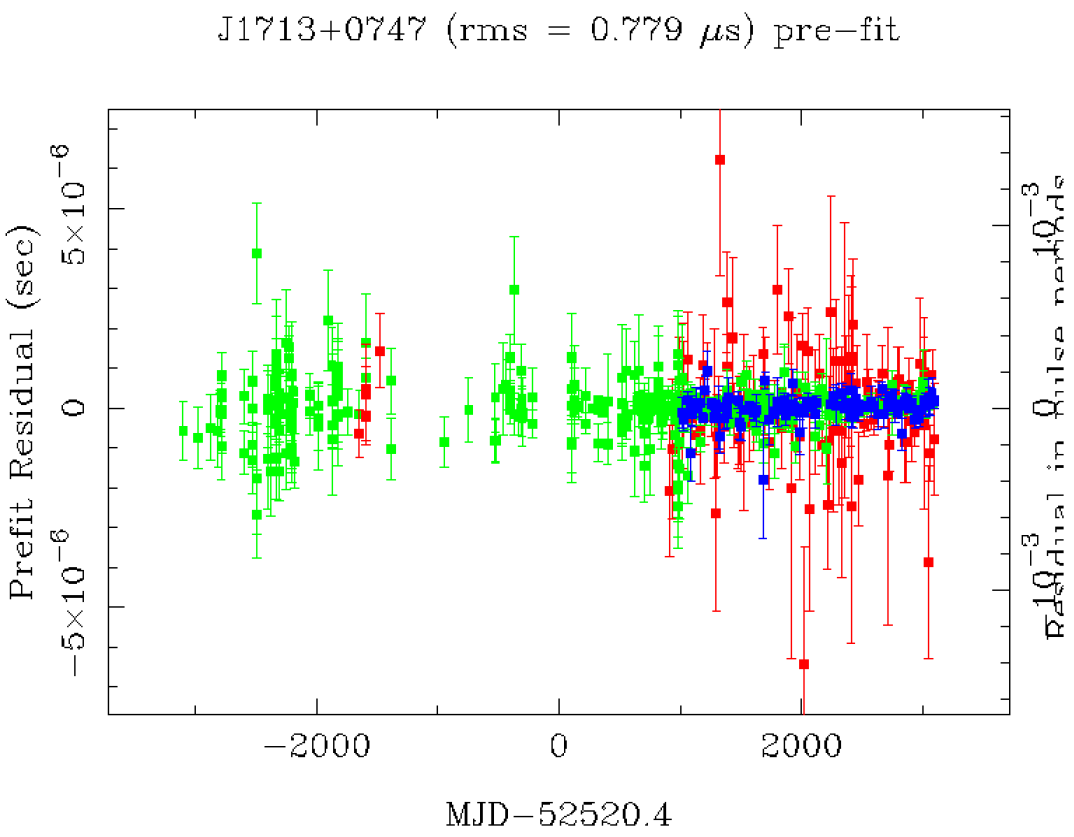

Figure 21: Simulation of data using the exact sampling and error bars from the PPTA J1713+0747 observations

3.15 REALISTIC ERROR BAR SIZES

It is possible to simulate realistic error bar sizes. This requires:

- Determination of the noise level using the radiometer equation
- Determination of the ToA uncertainty from the noise level and pulse profile shape.

The white noise level is calculated from the radiometer equation:

$$\sigma = \frac{(T_{sys} + T_{sky})}{G\sqrt{2\Delta\upsilon t_{obs}}}$$

where tobs is specified for a given observation. DeltaNu is specified for a given backend, Tsys and G for a specific receiver and T_sky for a particular pulsar and observing frequency.

This noise is added to an analytic, noise-free representation of the pulse profile at a given frequency (see Section XX). The analytic profile template is rescaled to equal the mean flux density provided for the specific pulsar:

<pulsars>

psr: name=J0613-0200, profileFile=[1400;0613_20cm.std], profileFile=[730;0613_50cm.std], profileFile=[3100;0613_10cm.std], flux=[1400;2.25],flux=[730;6.7], flux=[3100;0.45], diff_df=[1440;2.14e6], diff_ts=[1440;756],tsky=[740;5.85], tsky=[1400;1.07], tsky=[3100;0.14]
</pulsars>

Note that definitions of the profile file, flux density, tsky and the diffractive scintillation parameters that are to used at different observing frequencies.

It is also necessary to provide information about the receivers and backend instruments used:

<rcvr>
name: pks_mb
flo: 1230
fhi: 1530
tsys: 21
gain: 0.62
</rcvr>

<rcvr>
name: pks_10cm
flo: 2600
fhi: 3600
tsys: 35
gain: 0.9
</rcvr>

<rcvr>
name: pks_50cm
flo: 700
fhi: 764
tsys: 40
gain: 0.9
</rcvr>

<be>
name: pks_dfb
bw: 230
nbin: 1024
</be>

The schedule must define which receiver and backend to use for each observation:

<schedule>

name: sched1
observe: psr=J0613-0200,toaerr=radiometer,rcvr=pks_mb,be=pks_dfb,freq=1400
observe: psr=J0613-0200,toaerr=radiometer,rcvr=pks_10cm,be=pks_dfb,freq=3100
observe: psr=J0613-0200,toaerr=radiometer,rcvr=pks_50cm,be=pks_dfb,freq=730
</schedule>

By default the Taylor (1992) algorithm is run to determine the error bar size using the analytic template and the simulated profile. The resulting profiles are also made available for subsequent study.

The figure below shows the resulting ToA uncertainties for the three different simulated frequency bands:

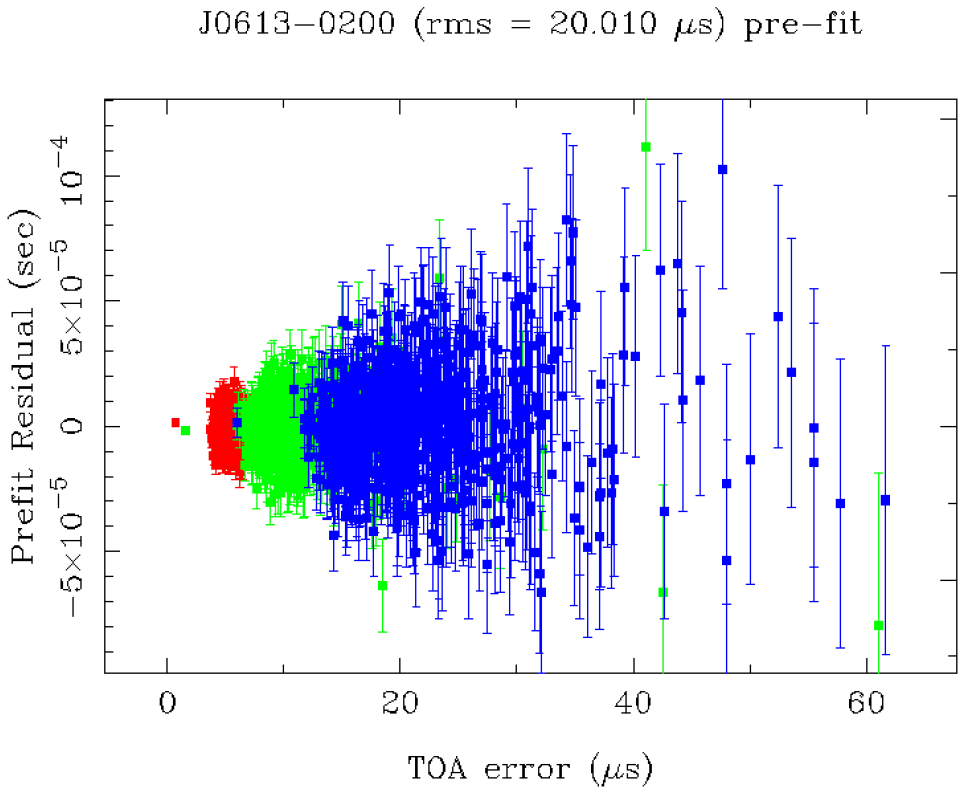

Figure 22: ToA uncertainties in three observing bands (red = 50cm, green = 20cm, blue = 10cm) without scintillation

The range of uncertainties for a given frequency band is not caused by scintillation (which, in this case, has not been simulated). Instead the scatter represents different ToA uncertainties from the Taylor algorithm for data sets with the same noise properties.

It is also possible to use an input file drawing random values from a list of measured ToA uncertainties. Such a file needs to be provided for each observation system.

3.16 SCINTILLATION

The scintillation models are based on measurements (or simulations) of the scintillation bandwidth (delta_nu) and timescale (ts).

3.16.1 Simple scintillation model

The simplest model is to determine the approximate number of scintles in the specified band and then assume that each scintle will change the flux following an exponential distribution. The final flux value is then obtained by summing over all the individual scintles.

3.16.2 Simulation of dynamic spectra

3.17 INSTRUMENTAL OFFSETS AND ARTEFACTS

Each backend instrument can have arbitrary phase offsets with respect to data from other backends. An example is:

```
<define>
name: sim8
nproc: 1
nreal: 1
t2exe: tempo2-dev
</define>

<pulsars>
psr: name=J0437-4715
psr: name=J1713+0747
psr: name=J1909-3744
</pulsars>

<obsRun>
name: pks
start: 53371
finish: 56000
tel: pks
sched: sched1
sampling: cadence=20
</obsRun>

<be>
name: be1
offset: mjd=54300,size=-1e-6
offset: mjd=55000,size=1e-6
</be>
```

<be>
name: be2
offset: mjd=54800,size=-1e-6
</be>

<schedule>
name: sched1
observe: psr=J0437-4715,toaerr=1e-7,freq=1400,be=be1
observe: psr=J1713+0747,toaerr=1e-7,freq=1400,be=be1
observe: psr=J1909-3744,toaerr=1e-7,freq=1400,be=be2
</schedule>

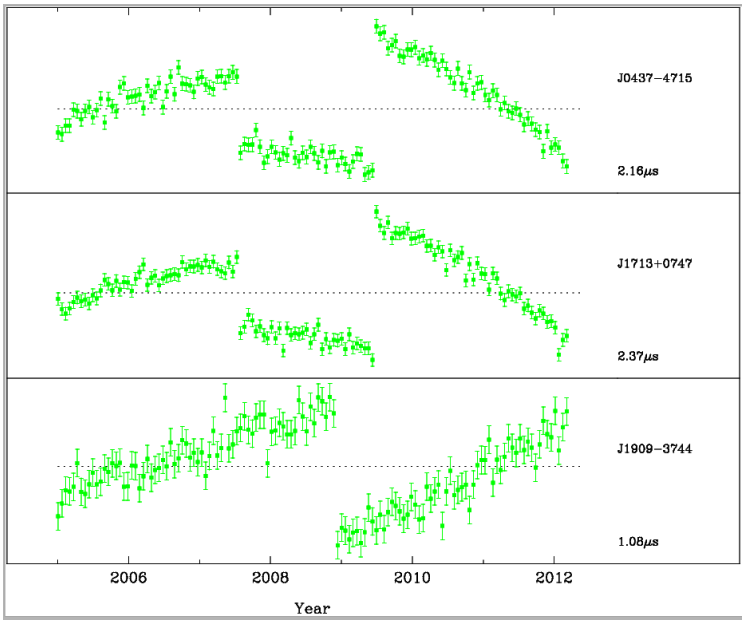

Figure 23: The addition of unphysically large instrumental jumps

It is also possible to add in extra noise caused by errors in the backend. The following example will add in 1us of Gaussian noise to each ToA (perhaps representing an error in the start time of the instrument).

<be>
name: be2
addNoise: 1e-6*ran(gaussian)
</be>

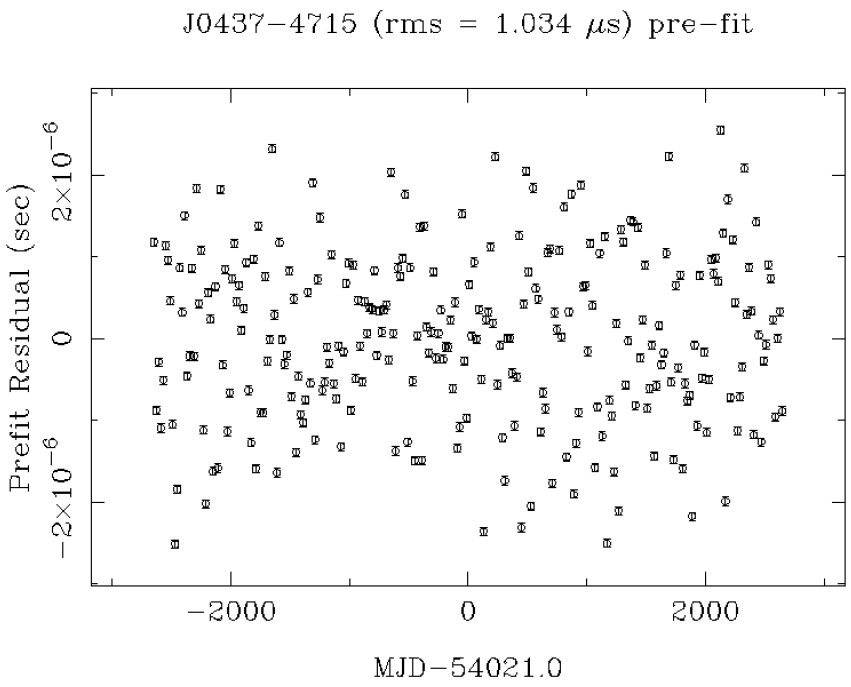

Figure 24: Simulation in which the backend introduces extra jitter into the residuals

In some cases it seems as if the backends can quantize the data. This can be simulated using (to produce 8 quantised levels at 50ns):

<be>
name: be2
addNoise: 50e-9*(ran(int;8)-4)
</be>

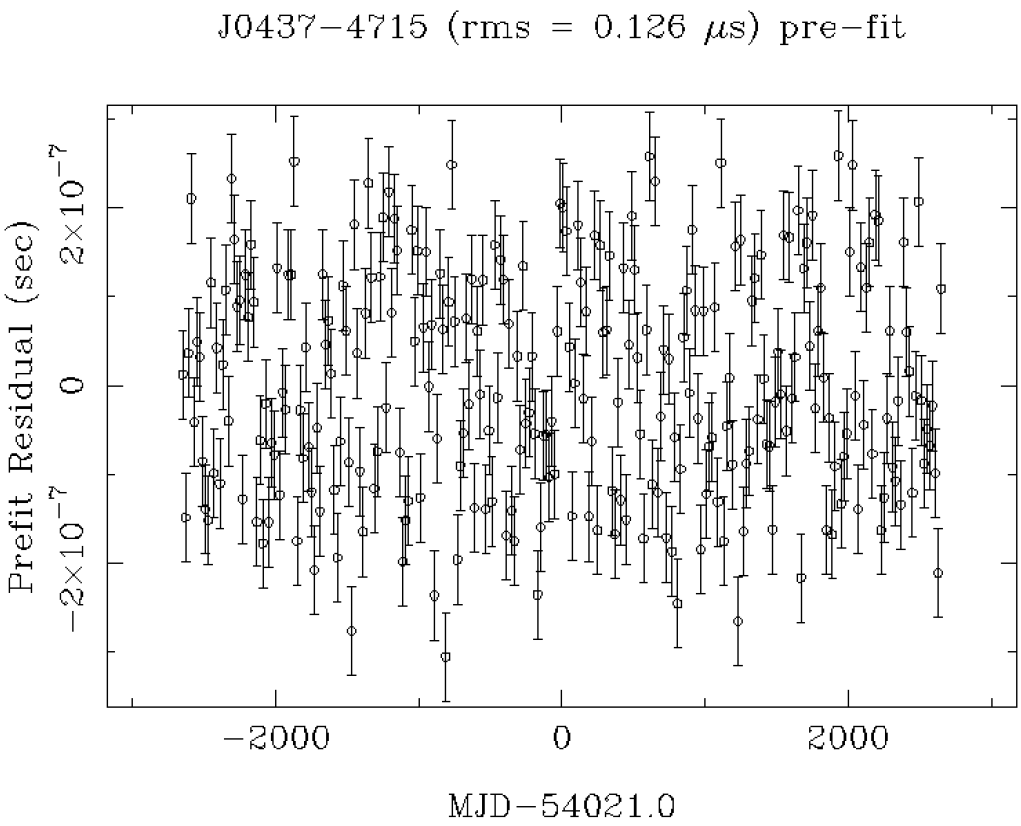

Figure 25: Simulation in which the backend adds quantised noise into the residuals

It is also common to find some outlier residuals, which cannot easily be explained. These could come from the receiver system, the backend instruments, errors in the clocks, or could even be a physical phenomenon affecting the pulsar. Such outliers can be simulated by defining the probability of an outlier and also the affect of the outlier in a schedule:

<schedule>
name: sched1
observe: psr=J0613-0200,toaerr=1e-6,freq=1400,outlier=[10e-3*ran(linear)-5e-3;ran(linear)>0.9]
</schedule>

In the example above, the probability of having an outlier is calculated by evaluating the expression “ran(linear)>0.9”. The size of the outlier is then (10e-3*ran(linear)-5e-3). This gives the result shown below:

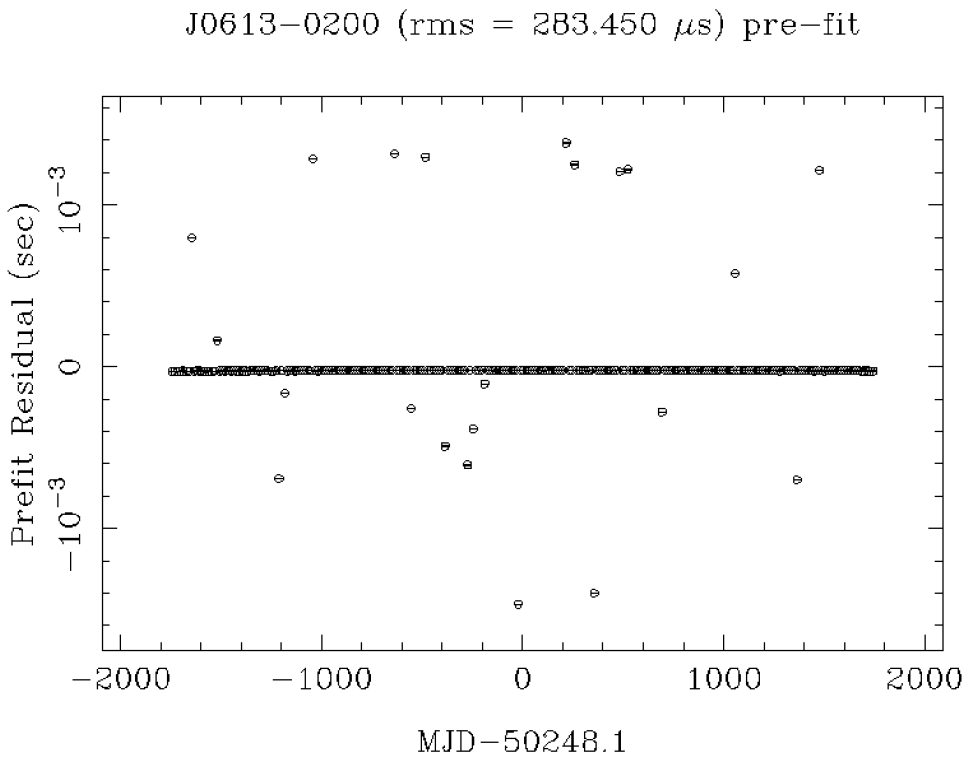

Figure 26: Adding outlier points into a dataset

3.18 LABELLING PULSARS

Within the input file pulsars are generally defined by their name. However, when pulsars have been simulated their names are unknown. In this case it is essential to label the pulsars.

3.19 SIMULATING PULSARS

Pulsars can be simulated as:

<pulsars>

createpsr: n=24,pos=isotropic(fixed;decj<20), p0=10**(log10(1.7)+ran(linear)*log10(4.0/1.7))*1e-3,p1=10**(-21+ran(linear)*2),dm=10+ran(linear)*100,pepoch=56000,label=new

</pulsars>

This definition will invent 24 (n=24) pulsars. Their positions will be randomly selected and their periods (p0), period derivatives (p1) and dispersion measures (dm) drawn from an analytically defined expression.

Describe options for pos. State which parameters are compulsory.

3.20 RFI

3.21 SIMULATING SUBINTS AND SUBCHANNELS

3.22 USING THE PROFILES

3.23 USING THE TEMPORARY OUTPUT FILES

4 PULSE TEMPLATES

For the simplest simulations it is not necessary to provide pulse templates. In such cases it is necessary to define the ToA uncertainties directly. For realistic simulations it is necessary to determine the ToA uncertainties from a pulsar profile template and knowledge of the receiver and backend parameters. The pulse templates are defined in the **ptime** format developed by G. Hobbs and S. Dai. It is therefore necessary to download the ptime software and create new templates for all pulsars that are to be simulated.

5 MONITORING THE RUNNING OF THE SOFTWARE

6 RUNNING ON A SUPERCOMPUTER

7 USING THE OUTPUT SIMULATIONS ON ANOTHER COMPUTER

- Show how to ensure the same T2runtime is used

8 EXPRESSION PARSER

Most of the parameters defined in the input file can be provided as a single number or as an expression. For instance to define the observing cadence to be exactly 20 days:

```
sampling: cadence=20
```

To define (for some reason) a simple mathematical equation

```
sampling: cadence=20+5*2
```

will make the cadence 30 days. More usefully is to use a random distribution:

```
sampling: cadence=20+5*ran(linear)
```

will choose sampling randomly (linearly) between 20 and 25 days. ran(linear) returns a random number between 0 and 1. To use a Gaussian distribution:

```
sampling: cadence=20+5*ran(gaussian)
```

will select the cadence to be between 15 and 25 for 1 sigma. It is also possible to list possible values in a file. The software will choose a random value out of the file (which must simply be a list of values – 1 per line):

```
sampling: cadence=flist(myFile.dat)
```

Note that the expressions can be included together, e.g.:

```
sampling: cadence=flist(myFile.dat)+5*ran(gaussian)
```

Most of the parameters are chosen once for each realization. However, a few parameters are re-evaluated more often:

Parameter	When evaluated
cadence	After each observing session

For a few parameters it is not clear whether they should be updated after each realization. These are:

- pos (when simulating pulsar positions it is often required that those positions stay constant between different realisations)

The user can select whether they should be held fixed or not by ….

9 KNOWN ISSUES

- Problem with clocks etc. if the corrections are large – can use TOASIM
- Can have issues with tropospheric corrections is simulated telescope is pointing into the ground. Use CORRECT_TROPOSPHERE N or use realistic rise and set times.

10 THE ALGORITHMS

10.1 THE PULSARS

10.1.1 Known parameters used for simulation

The set of pulsars to be included in the simulation need to be defined. Their parameters can be specified by the user or obtained from the ATNF pulsar catalogue. The simplest usage is to simply state their PSRJ names and the program obtains their parameters from the catalogue:

```
<pulsars>
 psr: J0437-4715
 psr: J0613-0200
</pulsars>
```

In this case a "psrcat –e" command is called. The software automatically removes the following parameters: RM, UNITS, EPHEM, CLK, START, FINISH, EPHVER. Each resulting parameter file automatically includes UNITS TCB. The CLK, EPHEM and other related parameters are defined elsewhere.

Inventing pulsars

10.1.2 Known parameters used in "observation parameter files"

The software provides a "parameter" file to the user to represent the best available parameter file for each pulsar. That parameter file need not be identical to the one used for the simulation (that represents the real universe). By default the "observation" parameter file is identical to the simulation file.

10.1.3 Unknown parameters

10.2 OBSERVATIONS

10.2.1 Receivers in the simulation

Each observatory can have a suite of receivers. These are defined using:

```
<rcvr>
id: <ID>
name: <LABEL>
freq: <Central frequency (MHz)>
bw: <BW (MHz)>
</rcvr>
```

How do we define dual frequency receivers?

10.2.2 Backend instruments at the observatory

An observatory usually has a suite of backend instruments. The PPTA often observes simultaneously with two or more backends. In order to simulate this the user has to define the individual backends available at the observatory:

```
<backend>
id: <ID>
name: <LABEL>
bw: <BW (MHz)>
</backend>
```

Note that the same backend can be used with different receivers. The bandwidth given here (BW) is the maximum bandwidth available. The exact choice of bandwidth used for a given system is calculated as the minimum of the receiver bandwidth and the instrument bandwidth.

Each observation is then "carried out" using a suite of backends. These are defined using:

```
<beSuite>
id: <ID>
backend: <ID of backend1>
backend: <ID of backend2>
…
</beSuite>
```

MUST THINK ABOUT 10/50CM SIMULTANEOUS OBSERVATIONS

10.2.3 Observing schedules

The entire simulation is controlled by a set of "obsRun" commands. These define the observations from a particular observing programme:

```
<obsRun>
name: <LABEL>
tel: <TEL_ID>
sched: <SCHED_ID>
start: <START (MJD)>
end: <END (MJD)>
cadence: <SESSION CADENCE (D)>
</obsRun>
```

If data from multiple telescopes are to be simulated then they can be described by multiple <obsRun> commands. They can also be used to describe changes in an observing program (i.e., when new pulsars are added into the sample, or the observing cadence changes). The <obsRun> defines when observing sessions occur. The observations that take place during each observing session is defined using a schedule:

```
<schedule>
id: <ID>
startLST: <LST start of schedule>
obs: <psr> <tobs (min)> <rcvrID> <beSuite ID> <whiteNoise level (ns)>
obs: <psr> <tobs (min)> <rcvrID> <beSuite ID> <whiteNoise level (ns)>
…
</schedule>
```

The "obs:" commands define which pulsar is being observed, for how long, and with which receiver and backend instruments.

MUST THINK BETTER HOW TO DEFINE THE WHITE NOISE

10.3 NOISE PROCESSES

10.3.1 Scintillation

10.3.2 DM variability

10.3.3 Radiometer noise

10.3.4 Jitter noise

10.3.5 Instrumental noise

10.3.6 Generic red noise

11 DEFINING THE OUTPUT

11.1 NUMBER OF REALISATIONS

12 SETTING UP THE SOFTWARE

12.1 TEMPO2 FILES

12.2 NUMBER OF PROCESSING NODES

13 ENSURING REPRODUCIBILITY

14 RUNNING THE SOFTWARE

14.1 COMPILING

14.2 MONITORING

The software can take a long time to finish. It is therefore necessary to be able to check the current status of the processing, be able to stop it as necessary and to be able to restart.

The scripts automatically run multiple processes. To stop these processes:

- touch <name>/scripts/status/stopScript

(all scripts should stop when they complete their current processing)

The current status of the processing (and to know which realisations have been completed):

- more <name>/scripts/status/runStat

15 EXAMPLE INPUT FILES

Label	Definition	Complete?
S1	20 pulsars, 100ns white noise, 14 day sampling	Y
S2	As S1 + GWB(1e-15)	Y
S3	As S1, but provide different time standard for processing	Y
S4	As S1, but provide different planetary ephemeris for processing	Y
S5	As S1, but use different solar wind model for processing	Y
S6	As S1, but have the output divided into 1 year chunks	-
S7	As S1 + uncorrelated red noise	Y
S8	As S1 + completely correlated red noise	Y

S9	As A1, but provide error in planet masses for processing	-
S10	Realistic simulation of the PPTA dr1 data	-
S11	Realistic simulation of the IPTA data	-
S12	Realistic prediction of data from FAST	-
S13	Realistic prediction of data from SKA	-
S14	Realistic prediction of new pulsars to be added to IPTA	-
S15	Realistic simulation of PPTA data with PAF + 10/50cm	-
S16	Realistic simulation of PPTA data with WBR	-`
S17	How to get a perfect HD curve?	-