Refactor notes

17/09/17

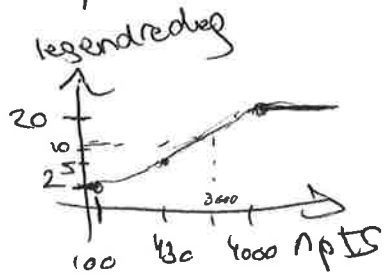
- leave whiten_allquarters as-is ... write in changes as iter_whiten_allquarters, would should just make it redundant.

legendre deg selection: somewhere btwn ~~4~~ 6 & 20 depending on npts.

10 days, 48 pts/day ≈ 480 pts.

6 days ≈ 420 pts.

2 days ≈ 100 pts

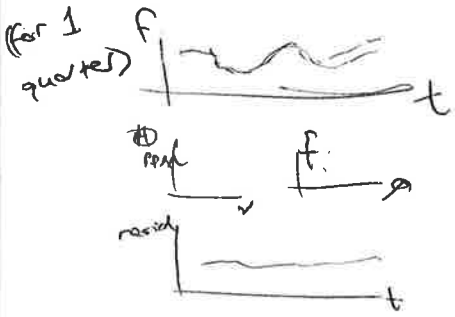


note residual becomes data for next iteration

factor of 2 in storage

Visualizer:

- Basically same as 3row plots ... but w/ iters in title so



TODO:

add a "redetrend" step (on f(t), 1 ≈ 20 legendre)

rms floor:

set at 0.25%

= 0.0005

Algorithm for removing all periodic components of EB variability, while keeping planet dips:

17/03/16

- For each quarter, remove (n=20 finite Legendre series – a medium-order polynomial) slow-varying f(t) variability, (Δ normalize to relative flux units).
- while RMS > 0.1% (SNR per transit of a ~4R⊕ CBP being ~1):
  - Stellingwerf phase dispersion minimiz^n (coarse freq bins) & get coarse period.
    ↳ repeat over narrow bins centered on peak signal.
    Select ★ period, which we will "whiten" at.
  - Phase-fold at period and fit (finite-order Legendre series again. Order can be determined by cross-validation, or AIC/BIC, to avoid overfitting & underfitting. It should be Low enough that it will gloss over transits [too sharp].
  - Subtract fit. Compute new RMS.

(n.b. astrobase/varbase/signals.gls_whiten does a similar thing, but with GLS ~1/2 then fit out nbestperiods)

The main assumption this makes is that removing the periodic components (10³ ppm) of the EB will be a sufficient way to drive everything below RMS of 0.1%. (500ppm)
(If we're optimistic & write the program well, possibly even to <0.05%, which would really put our completeness at "beyond a doubt" levels.).

While aperiodic variability (attributed to starspots / magnetic activity) could be important, fitting models to it is generally trickier.
(See: ARMA, ARIMA, ARFIMA, & think further about how autoregression-"does the pwarive EB signal here match whatever is in the data there?"-
could be a path forward there...
note similarly that 'george' might be faster for GP regression than the sklearn implementation I tried).

Most importantly, all of the above fancy statistical crap MAY NOT BE NECESSARY (ideally, it WILL NOT be necessary) for the main result – completeness for at least 4R⊕ CBPs about big pulsators – to hold!.

Stellingwerf (1978) Phase Dispersion Minimize?

- Nonparametric whitening: for a proposed period, phase-fold & bin over N bins. Compute mean & std dev^n for each bin. These are related to Ⓝ, a statistic which becomes small (negative?) for the lowest std dev^s of all proposed periods. (n.b. the binned std dev^n / variance is compared to the non-phased variance — the ratio will be ≈1 for a false period, and ≪1 for a true period).

Mags $\vec{x}$, times $\vec{t}$ (x_i, t_i) for $i = \{1, \ldots, N\}$.

$$\sigma^2 = \frac{\sum (x_i - \bar{x})^2}{N-1} \quad (1)$$

(σ^2: variance of mags)

Choose M distinct samples, with variances s_j^2 $(j = \{1, \ldots, M\})$. Say each contains n_j data points. The variance for all the samples is

$$s^2 = \frac{\sum (n_j - 1) s_j^2}{\sum n_j - M} \quad (2)$$

Ⓑ Trial period Π, has phase vector $\vec{\Phi}$: $\Phi_i = t_i/\Pi - \lfloor t_i/\Pi \rfloor$ (floor, or "take integer part").

Divide $\vec{\Phi}$ into bins (e.g., 10 or 100). Compute

$$\Theta = \frac{s^2}{\sigma^2}.$$

Note Ⓝ ≈ 1 for false periods, but is a local minimum for true periods.

But how significant are these peaks? Schwarzenberg-Czerny (1997) notes that comparing Θ with the "Fischer-Snedecor F statistic" isn't quite right — in fact Θ follows a Beta distribution.

(Schwarzenberg-Czerny also notes that "the high-performance Fourier series method based on orthogonal projectors" weakens the use of PDM).

(He also notes that the step-function binning has some undesirable properties).

↑ which can be circumvented w/ linear fits)

17/03/16

we want a low pass filter.

Butterworth filter has gain $G(\omega)$ at n^{th} order of

$$G^2(\omega) = |H(j\omega)|^2 = \frac{G_0^2}{1 + \left(\frac{j\omega}{j\omega_c}\right)^{2n}}$$

n = filter order
ω_c = cutoff freq (n -3 dB freq)
G_0 = DC gain.

as $n \to \infty$, the gain becomes a rectangle func^n.

Scipy's implementation...

N.b. digital filters operate on discrete data, analog continuous (?)

Gives numerator 'b' & denominator 'a' of an IIR (or FIR) filter.

"IIR": infinite impulse response.

$$y[n] = \frac{1}{a_0}\left(\sum_{i=0}^{P} b_i\, x[n-i] - \sum_{j=1}^{Q} a_j\, y[n-j]\right)$$

$y[n]$: output signal

$x[n]$: input signal

a_n: feedback filter coeffs

b_n: feedforward filter coeffs

Q: feedback order

P: feedforward order

Perhaps unsurprisingly, in practice this looks like not so good.

Maybe what we want is a finite impulse response filter constructed from windows?

- what difference does having planets make in frequency domain?

- Another route for improvement:

better initial detrending. When there are systematics there, they come into the fold.

target here: 0.1% RMS.

OR just do iterative PDM (See if we get anywhere).

Something like dipsearchplot, but for closer checking of why(not) confounding 17/03/15

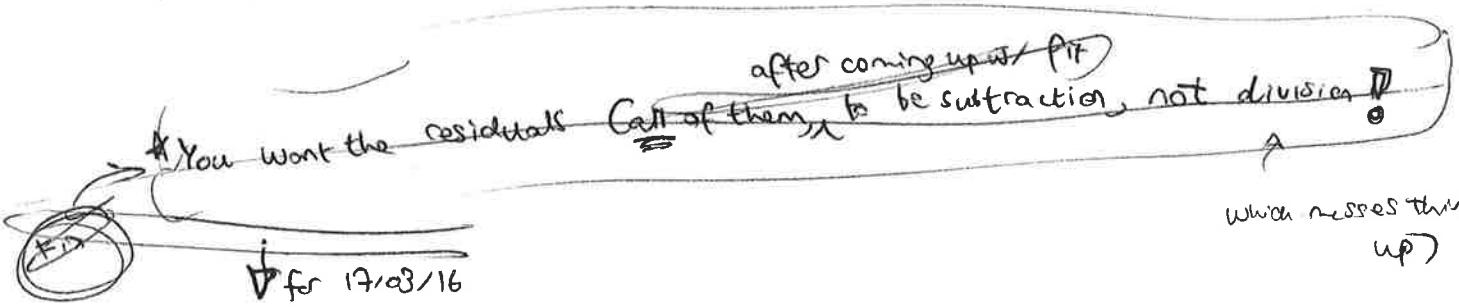
good example EB subtraction cases:

• 4273411, SAP, $P = 1/2.16$ ⇒ 0.000625 = 0.0625% dips.

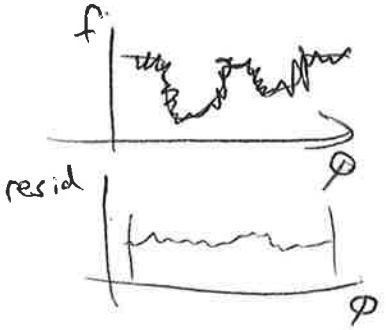
• 4554004, only 2 quarters!
↳ short-term periodicity, with a BEAT.
(≲ 1 day)

• 4660997: more detached, WIF bad fit to begin ✓.

• 4850894: short-term periodicity at ≲1 day level?



in phase-folded plots, the amount of "fuzz" around the fit all will show up...



So rather than iteratively whitening by PDM → fit → subtracting, ~~rediscovering the~~ rerunning PDM, refitting, resubtracting, ...)

* use cross-validation (w/in a/ quarter? or inter-quarter...) to select preferred smoothing scale...
[should ideally be doing this at EVERY fitting step!
(implement it in the "EB signal subtraction fitting" subroutine).

17/03/16:
* Butterworth filter
• Iterative whitening on PDM periods

17/03/15

Gaussian process ≡ collection of random variables in parameter space, any subset of which can be specified by a joint Gaussian distrib'n.

To specify, need
1) mean
2) covariance function ("kernel").

Different kernels (covariance functions) have different "hyperparameters".

E.g. the squared exponential covariance function:

$$Cov(x_1, x_2; h) = \exp\left(-\frac{(x_1 - x_2)^2}{2h^2}\right)$$

(h: hyperparam.)

If you have periodic funcns, consider the ExpSineSquared kernel: has length parameter l and periodicity param $P > 0$

$$Cov(x_1, x_2, l, P) \equiv k(x_i, x_j) = \exp\left[-\frac{2}{l}\sin\left(\frac{\pi}{P}\underbrace{d(x_i, x_j)}_{|x_i - x_j|, \text{ or } \sqrt{(x_i - x_j)^2}}\right)\right]$$

---

can try out above, OR just try

- iterative whitening.
- & remove data near gaps

meaning: iterative spline fitting! per F Dai's code.

↳ Implement as iterative whitening, w/ all same options (default: spline)

where are failures?

- 10905804
- ~~10920806~~
- 10965091
- 11036692
- 10032392: 1% dips, but periodicity (shorter P) drowns out

0.