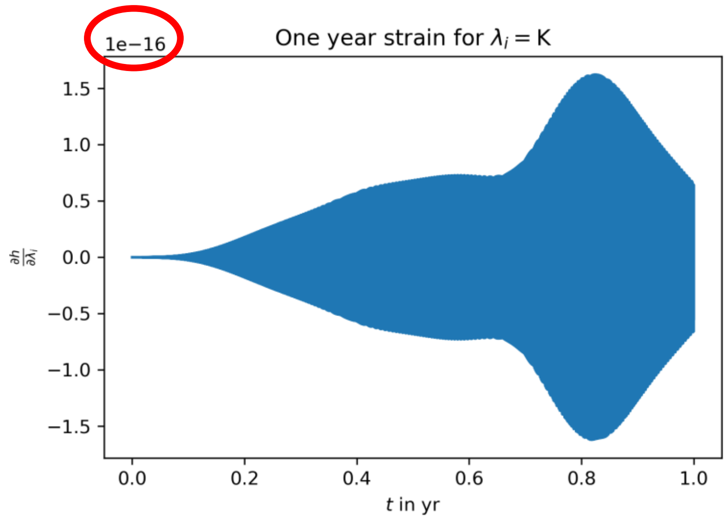
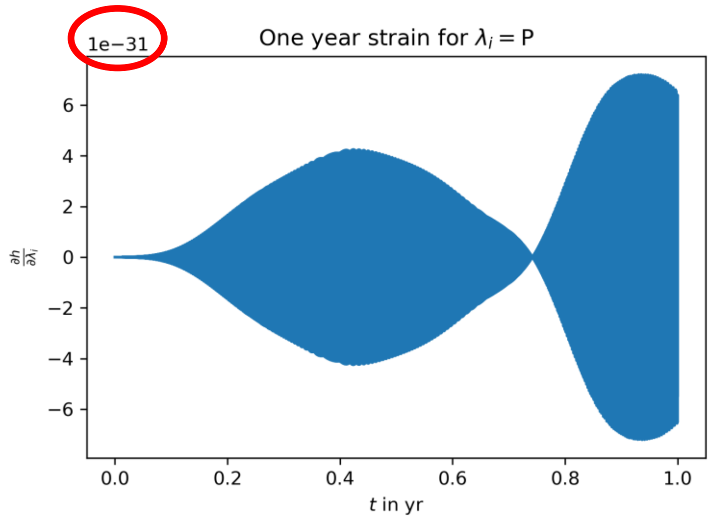
Detecting Exoplanets and recovering Parameters

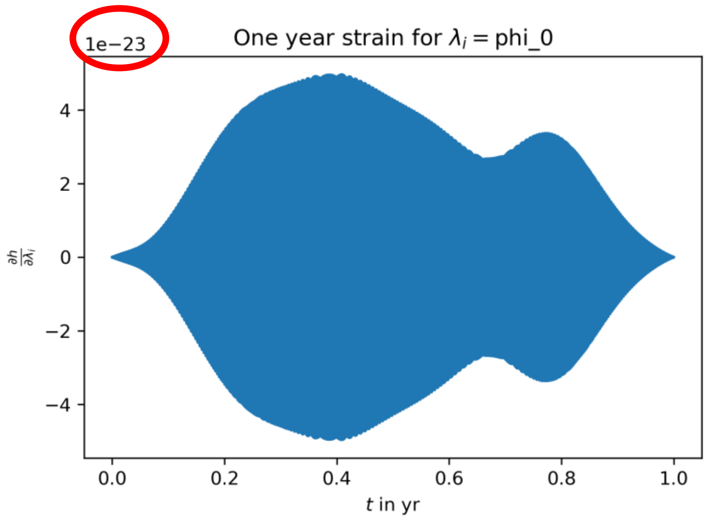
A short recap of my work in week 7

Points of interest

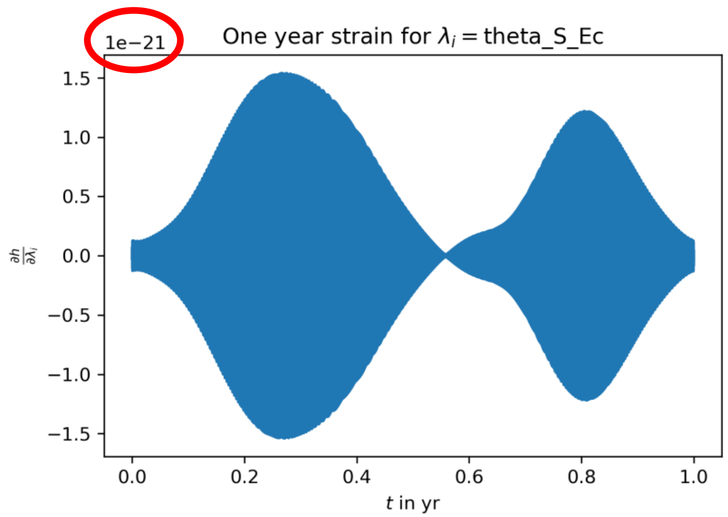
1. Problems in integration
2. Analytic derivations
3. Result for LISA
4. Regards on exoplanet constraints in Milky Way
5. IceGiant mission
6. A short proof of the scaling of exoplanet parameters with mass

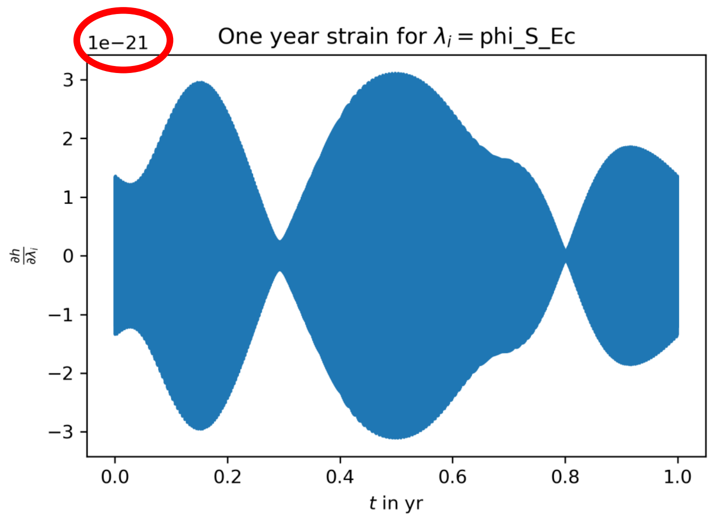


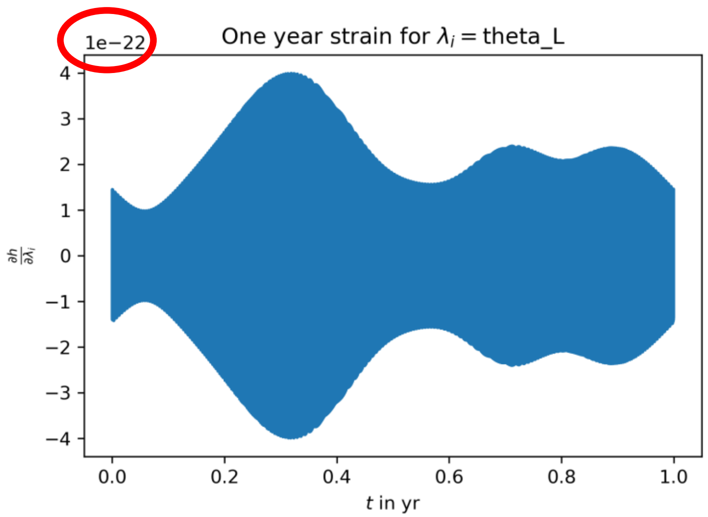


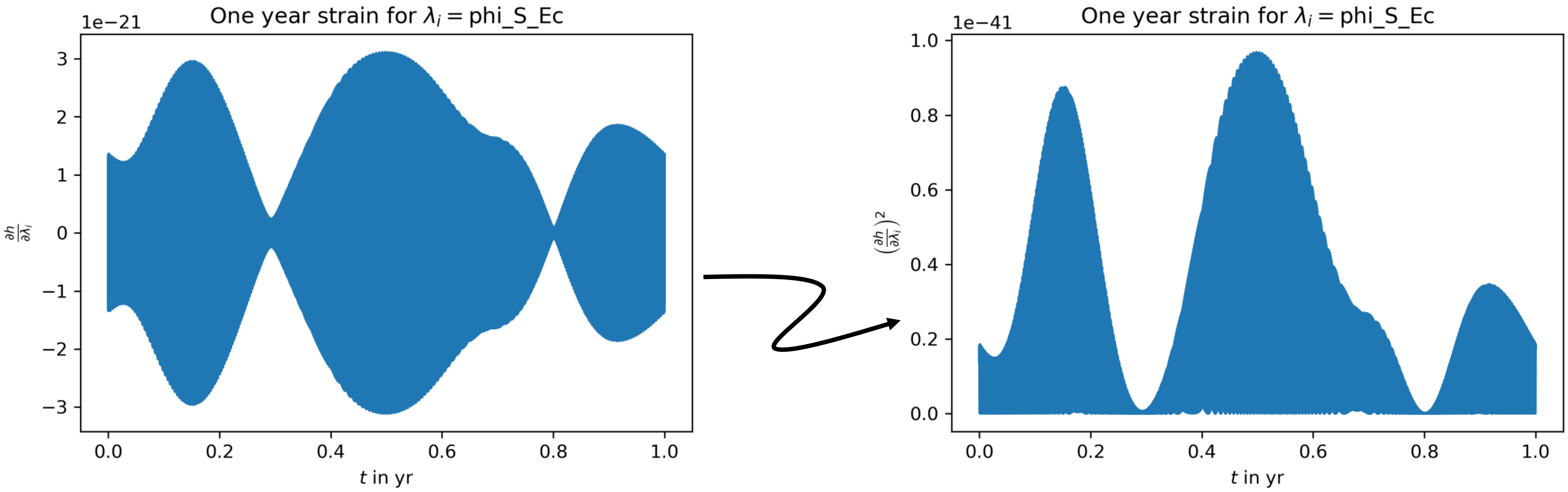


The derivatives $\frac{\partial h_\alpha}{\partial \lambda_i}$ look good, compare the analytic derivatives of the exoplanet par‘s (above) with the numerical derivatives of the position par‘s (below) – same structure: envelope * oscillating function









Numerical difficulties in the integration come from the product $\frac{\partial h_\alpha}{\partial \lambda_i}(t)\frac{\partial h_\alpha}{\partial \lambda_j}(t)$:

For amplitudes of $\frac{\partial h_\alpha}{\partial \lambda_i} \sim O(10^{-20})$ we get time dependant amplitudes $(\frac{\partial h_\alpha}{\partial \lambda_i})^2 \sim O(10^{-40})$ and this will strongly hinder the integration! (Machine epsilon for doubles: 2e-16)

Quick fix: $\int_0^{T_0} dt\, \frac{\partial h_\alpha}{\partial \lambda_i}(t)\frac{\partial h_\alpha}{\partial \lambda_j}(t) = 10^{-40} \times \int_0^{T_0} dt (\frac{\partial h_\alpha}{\partial \lambda_i}(t) \times 10^{20}) \times (\frac{\partial h_\alpha}{\partial \lambda_j}(t) \times 10^{20})$

1. Problems in integration

We compute a 9x9 symmetric matrix $\Gamma_{ij} \propto \left[\int_0^{T_0} dt \, \frac{\partial h_\alpha}{\partial \lambda_i}(t) \frac{\partial h_\alpha}{\partial \lambda_j}(t)\right]_{ij}$

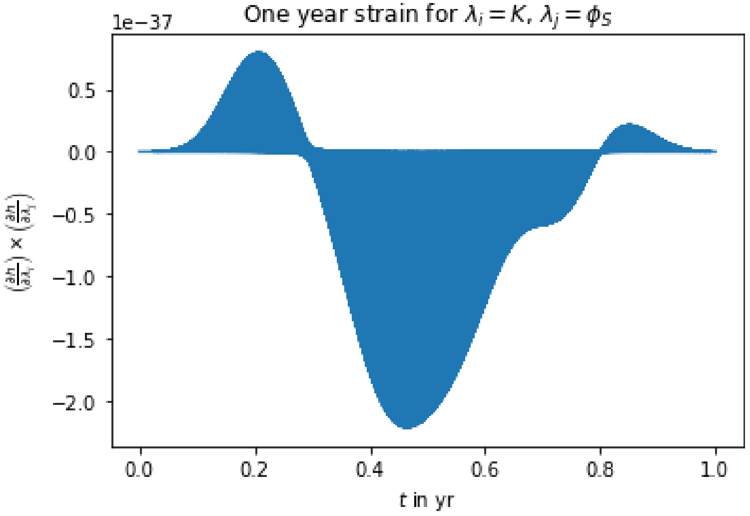
We know, diagonal elements Γ_{ii} will be in phase, so the integrand will go like $A^2 \sin^2 \omega t$ and we find for diagonal elements a result which goes roughly like $\Gamma_{ii} \propto \frac{1}{2}\int_0^{T_0} A^2(t)dt + O\left(\frac{A_0}{4\omega}\right) \approx \frac{1}{2} {A_0}^2 T_0 > 0$ with A_0 some representative value of the amplitude

A new problem has approached

We compute a 9x9 symmetric matrix $\Gamma_{ij} \propto \left[\int_0^{T_0} dt \, \frac{\partial h_\alpha}{\partial \lambda_i}(t) \frac{\partial h_\alpha}{\partial \lambda_j}(t)\right]_{ij}$

On the other hand, off-diagonal elements will go as $A_i A_j \sin \omega_i t \sin \omega_j t = \frac{A_i A_j}{2}(\cos(\omega_i - \omega_j)\, t \cos(\omega_i + \omega_j)\, t)$ and if they are uncorrelated, we'll find over long observations $\Gamma_{ij} \approx 0$

<u>**Question:**</u> What does *close to zero* mean for $\frac{1}{2} A_0{}^2 T_0 \sim O(10^{-40+7})$? Where do we make the cut?



A short recap on numerical integration

scipy.integrate calls QUADPACK from Fortran and then using a Clenshaw-Curtis method which uses Chebyshev moments computes the integral

Two parameters we can play with: epsabs and epsrel

The numerical integral `result` is returned if for the actual integral `i`

```
abs(i-result) <= max(epsabs, epsrel*abs(i))
```

Which can be estimated analytically

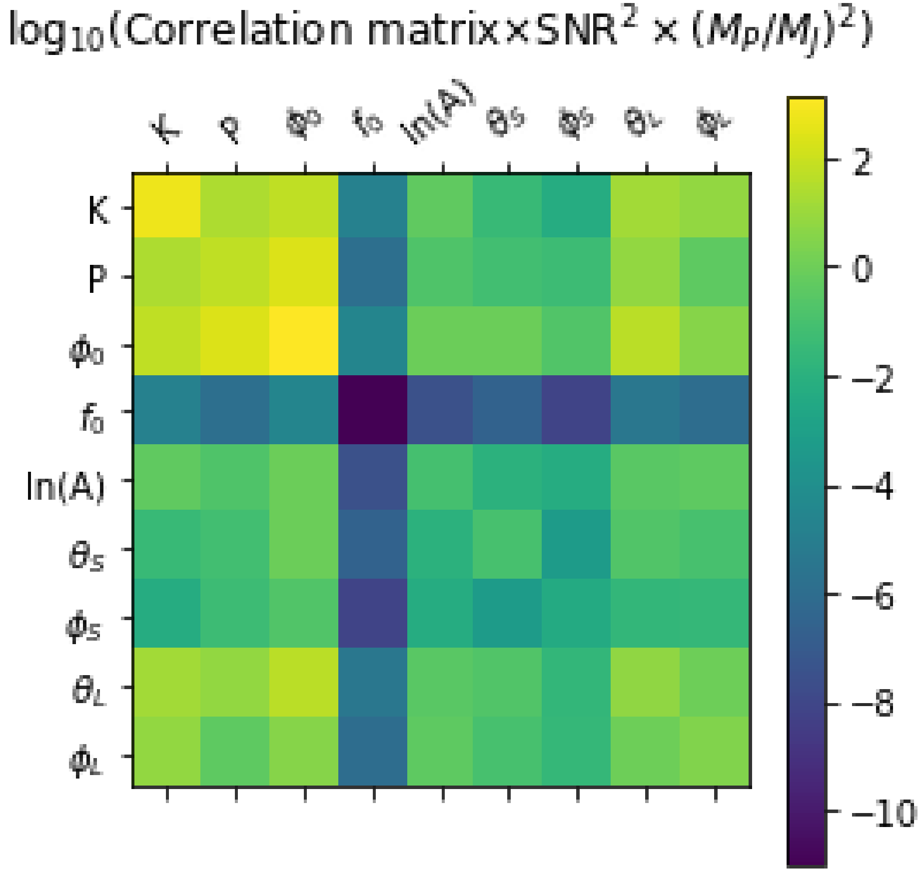
My trick for quicker computations: Set epsrel at 1.49e-2 and then set epsabs for off-diagonal integrals as multiple of geometric mean $\sqrt{\Gamma_{ii} \times \Gamma_{jj}}$ ×1.49e-3 -> don't waste time on irrelevant integrals

Findings

Looking at the correlation matrix for the 9 parameters of interest:

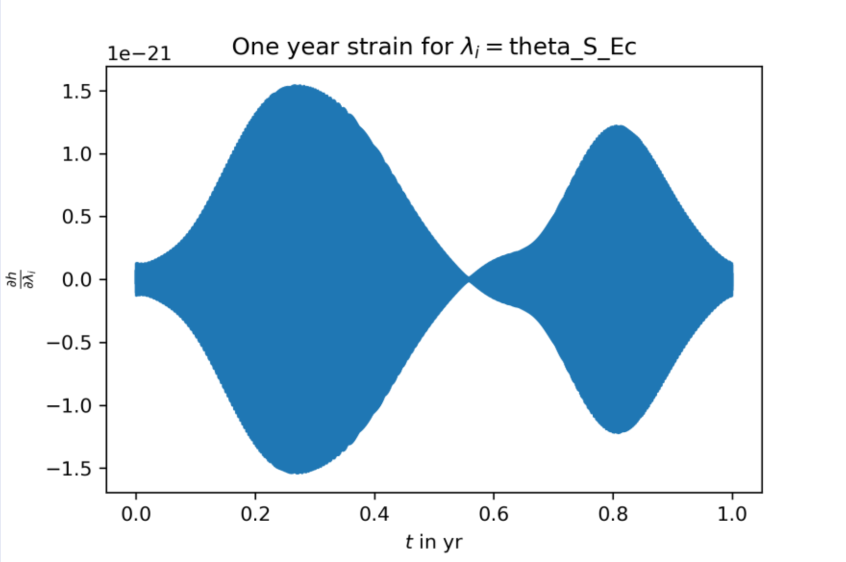
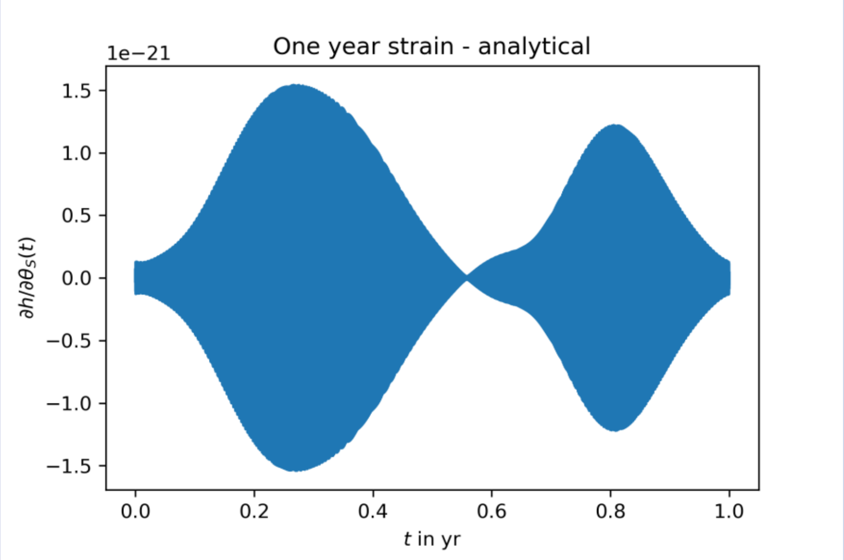
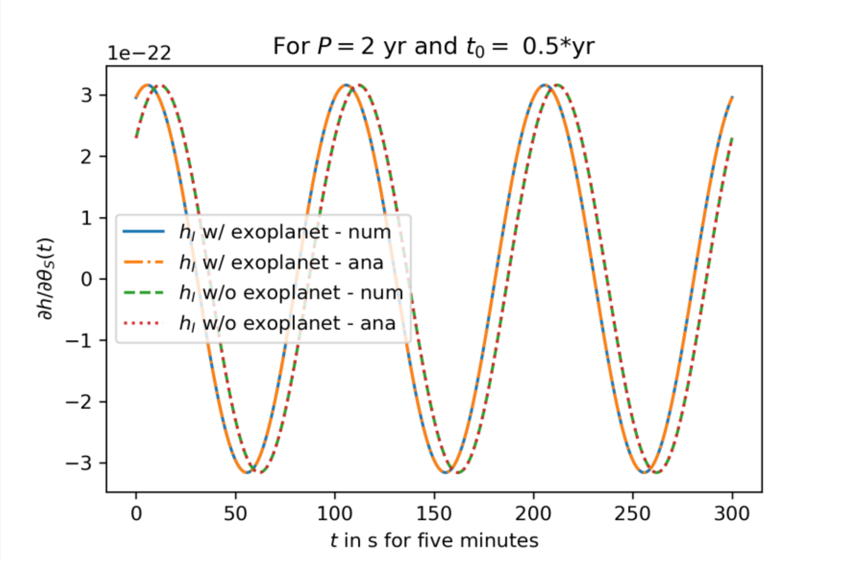
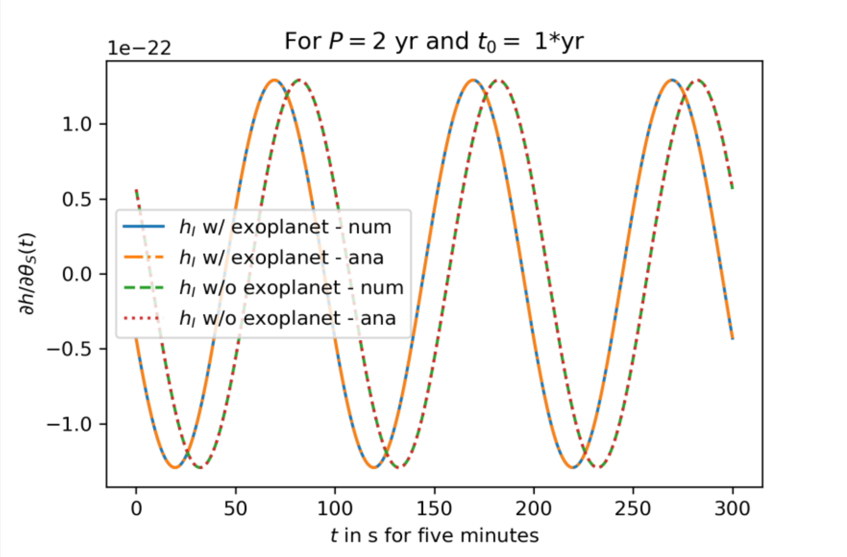
I again used the same position as Cutler, f_0 = 10 mHz and P = 2 yr

We can easily discard the f_0 fit, as the determination of the GW frequency isn‘t very problematic



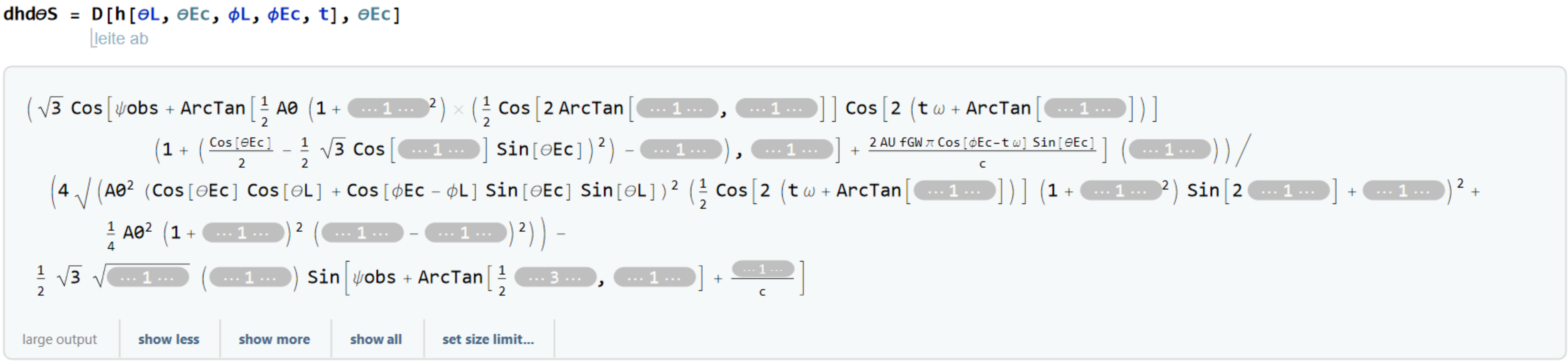
```
(np.sqrt(3)*(-(np.cos(self.theta_L)*np.cos(self.phi_S_Ec)*np.sin(self.theta_S_Ec)) + np.cos(self.theta_S_Ec)*np.cos(self.phi_L)*np.sin(self.theta_L))*np.sin(t*omega_E))/2.,
np.cos(self.theta_L)/2. - (np.sqrt(3)*np.cos(self.phi_L - t*omega_E)*np.sin(self.theta_L))/2. -
(np.cos(self.theta_S_Ec)/2. - (np.sqrt(3)*np.cos(self.phi_S_Ec - t*omega_E)*np.sin(self.theta_S_Ec))/2.)*(np.cos(self.theta_S_Ec)*np.cos(self.theta_L) + np.cos(self.phi_S_Ec - self.phi_L)*np.sin(self.theta_S_E
np.sin(2*(t*omega_E + ArcTan(-2*np.sin(self.theta_S_Ec)*np.sin(self.phi_S_Ec - t*omega_E),np.sqrt(3)*np.cos(self.theta_S_Ec) + np.cos(self.phi_S_Ec - t*omega_E)*np.sin(self.theta_S_Ec)))) -
(1 + (np.cos(self.theta_S_Ec)/2. - (np.sqrt(3)*np.cos(self.phi_S_Ec - t*omega_E)*np.sin(self.theta_S_Ec))/2.)**2)*
((-2*np.sin(self.theta_S_Ec)*(np.cos(self.theta_S_Ec)*np.cos(self.phi_S_Ec - t*omega_E) - np.sqrt(3)*np.sin(self.theta_S_Ec))*np.sin(self.phi_S_Ec - t*omega_E))/
((np.sqrt(3)*np.cos(self.theta_S_Ec) + np.cos(self.phi_S_Ec - t*omega_E)*np.sin(self.theta_S_Ec))**2 + 4*np.sin(self.theta_S_Ec)**2*np.sin(self.phi_S_Ec - t*omega_E)**2) -
(2*np.cos(self.theta_S_Ec)*(-(np.sqrt(3)*np.cos(self.theta_S_Ec)) - np.cos(self.phi_S_Ec - t*omega_E)*np.sin(self.theta_S_Ec))*np.sin(self.phi_S_Ec - t*omega_E))/
((np.sqrt(3)*np.cos(self.theta_S_Ec) + np.cos(self.phi_S_Ec - t*omega_E)*np.sin(self.theta_S_Ec))**2 + 4*np.sin(self.theta_S_Ec)**2*np.sin(self.phi_S_Ec - t*omega_E)**2))*
np.sin(2*ArcTan(-0.5*(np.sin(self.theta_S_Ec)*np.sin(self.theta_L)*np.sin(self.phi_S_Ec - self.phi_L)) -
(np.sqrt(3)*np.cos(t*omega_E)*(np.cos(self.theta_L)*np.sin(self.theta_S_Ec)*np.sin(self.phi_S_Ec) - np.cos(self.theta_S_Ec)*np.sin(self.theta_L)*np.sin(self.phi_L)))/2. -
(np.sqrt(3)*(-(np.cos(self.theta_L)*np.cos(self.phi_S_Ec)*np.sin(self.theta_S_Ec)) + np.cos(self.theta_S_Ec)*np.cos(self.phi_L)*np.sin(self.theta_L))*np.sin(t*omega_E))/2.,
np.cos(self.theta_L)/2. - (np.sqrt(3)*np.cos(self.phi_L - t*omega_E)*np.sin(self.theta_L))/2. -
(np.cos(self.theta_S_Ec)/2. - (np.sqrt(3)*np.cos(self.phi_S_Ec - t*omega_E)*np.sin(self.theta_S_Ec))/2.)*(np.cos(self.theta_S_Ec)*np.cos(self.theta_L) + np.cos(self.phi_S_Ec - self.phi_L)*np.sin(self.theta_S_E
np.sin(2*(t*omega_E + ArcTan(-2*np.sin(self.theta_S_Ec)*np.sin(self.phi_S_Ec - t*omega_E),np.sqrt(3)*np.cos(self.theta_S_Ec) + np.cos(self.phi_S_Ec - t*omega_E)*np.sin(self.theta_S_Ec)))))))/
(4.*np.sqrt(self.a0**2*(np.cos(self.theta_S_Ec)*np.cos(self.theta_L) + np.cos(self.phi_S_Ec - self.phi_L)*np.sin(self.theta_S_Ec)*np.sin(self.theta_L))**2*
((np.cos(2*(t*omega_E + ArcTan(-2*np.sin(self.theta_S_Ec)*np.sin(self.phi_S_Ec - t*omega_E),np.sqrt(3)*np.cos(self.theta_S_Ec) + np.cos(self.phi_S_Ec - t*omega_E)*np.sin(self.theta_S_Ec))))*
(1 + (np.cos(self.theta_S_Ec)/2. - (np.sqrt(3)*np.cos(self.phi_S_Ec - t*omega_E)*np.sin(self.theta_S_Ec))/2.)**2)*
np.sin(2*ArcTan(-0.5*(np.sin(self.theta_S_Ec)*np.sin(self.theta_L)*np.sin(self.phi_S_Ec - self.phi_L)) -
(np.sqrt(3)*np.cos(t*omega_E)*(np.cos(self.theta_L)*np.sin(self.theta_S_Ec)*np.sin(self.phi_S_Ec) - np.cos(self.theta_S_Ec)*np.sin(self.theta_L)*np.sin(self.phi_L)))/2. -
(np.sqrt(3)*(-(np.cos(self.theta_L)*np.cos(self.phi_S_Ec)*np.sin(self.theta_S_Ec)) + np.cos(self.theta_S_Ec)*np.cos(self.phi_L)*np.sin(self.theta_L))*np.sin(t*omega_E))/2.,
np.cos(self.theta_L)/2. - (np.sqrt(3)*np.cos(self.phi_L - t*omega_E)*np.sin(self.theta_L))/2. -
(np.cos(self.theta_S_Ec)/2. - (np.sqrt(3)*np.cos(self.phi_S_Ec - t*omega_E)*np.sin(self.theta_S_Ec))/2.)*(np.cos(self.theta_S_Ec)*np.cos(self.theta_L) + np.cos(self.phi_S_Ec - self.phi_L)*np.sin(self.theta_S_E
np.cos(2*ArcTan(-0.5*(np.sin(self.theta_S_Ec)*np.sin(self.theta_L)*np.sin(self.phi_S_Ec - self.phi_L)) -
(np.sqrt(3)*np.cos(t*omega_E)*(np.cos(self.theta_L)*np.sin(self.theta_S_Ec)*np.sin(self.phi_S_Ec) - np.cos(self.theta_S_Ec)*np.sin(self.theta_L)*np.sin(self.phi_L)))/2. -
(np.sqrt(3)*(-(np.cos(self.theta_L)*np.cos(self.phi_S_Ec)*np.sin(self.theta_S_Ec)) + np.cos(self.theta_S_Ec)*np.cos(self.phi_L)*np.sin(self.theta_L))*np.sin(t*omega_E))/2.,
np.cos(self.theta_L)/2. - (np.sqrt(3)*np.cos(self.phi_L - t*omega_E)*np.sin(self.theta_L))/2. -
(np.cos(self.theta_S_Ec)/2. - (np.sqrt(3)*np.cos(self.phi_S_Ec - t*omega_E)*np.sin(self.theta_S_Ec))/2.)*(np.cos(self.theta_S_Ec)*np.cos(self.theta_L) + np.cos(self.phi_S_Ec - self.phi_L)*np.sin(self.theta_S_E
(np.cos(self.theta_S_Ec)/2. - (np.sqrt(3)*np.cos(self.phi_S_Ec - t*omega_E)*np.sin(self.theta_S_Ec))/2.)*
np.sin(2*(t*omega_E + ArcTan(-2*np.sin(self.theta_S_Ec)*np.sin(self.phi_S_Ec - t*omega_E),np.sqrt(3)*np.cos(self.theta_S_Ec) + np.cos(self.phi_S_Ec - t*omega_E)*np.sin(self.theta_S_Ec)))))**2 +
(self.a0**2*(1 + (np.cos(self.theta_S_Ec)*np.cos(self.theta_L) + np.cos(self.phi_S_Ec - self.phi_L)*np.sin(self.theta_S_Ec)*np.sin(self.theta_L))**2)**2*
((np.cos(2*ArcTan(-0.5*(np.sin(self.theta_S_Ec)*np.sin(self.theta_L)*np.sin(self.phi_S_Ec - self.phi_L)) -
(np.sqrt(3)*np.cos(t*omega_E)*(np.cos(self.theta_L)*np.sin(self.theta_S_Ec)*np.sin(self.phi_S_Ec) - np.cos(self.theta_S_Ec)*np.sin(self.theta_L)*np.sin(self.phi_L)))/2. -
(np.sqrt(3)*(-(np.cos(self.theta_L)*np.cos(self.phi_S_Ec)*np.sin(self.theta_S_Ec)) + np.cos(self.theta_S_Ec)*np.cos(self.phi_L)*np.sin(self.theta_L))*np.sin(t*omega_E))/2.,
np.cos(self.theta_L)/2. - (np.sqrt(3)*np.cos(self.phi_L - t*omega_E)*np.sin(self.theta_L))/2. -
(np.cos(self.theta_S_Ec)/2. - (np.sqrt(3)*np.cos(self.phi_S_Ec - t*omega_E)*np.sin(self.theta_S_Ec))/2.)*(np.cos(self.theta_S_Ec)*np.cos(self.theta_L) + np.cos(self.phi_S_Ec - self.phi_L)*np.sin(self.theta_S_E
np.cos(2*(t*omega_E + ArcTan(-2*np.sin(self.theta_S_Ec)*np.sin(self.phi_S_Ec - t*omega_E),np.sqrt(3)*np.cos(self.theta_S_Ec) + np.cos(self.phi_S_Ec - t*omega_E)*np.sin(self.theta_S_Ec))))*
(1 + (np.cos(self.theta_S_Ec)/2. - (np.sqrt(3)*np.cos(self.phi_S_Ec - t*omega_E)*np.sin(self.theta_S_Ec))/2.)**2))/2. -
(np.cos(self.theta_S_Ec)/2. - (np.sqrt(3)*np.cos(self.phi_S_Ec - t*omega_E)*np.sin(self.theta_S_Ec))/2.)*
np.sin(2*ArcTan(-0.5*(np.sin(self.theta_S_Ec)*np.sin(self.theta_L)*np.sin(self.phi_S_Ec - self.phi_L)) -
(np.sqrt(3)*np.cos(t*omega_E)*(np.cos(self.theta_L)*np.sin(self.theta_S_Ec)*np.sin(self.phi_S_Ec) - np.cos(self.theta_S_Ec)*np.sin(self.theta_L)*np.sin(self.phi_L)))/2. -
(np.sqrt(3)*(-(np.cos(self.theta_L)*np.cos(self.phi_S_Ec)*np.sin(self.theta_S_Ec)) + np.cos(self.theta_S_Ec)*np.cos(self.phi_L)*np.sin(self.theta_L))*np.sin(t*omega_E))/2.,
np.cos(self.theta_L)/2. - (np.sqrt(3)*np.cos(self.phi_L - t*omega_E)*np.sin(self.theta_L))/2. -
(np.cos(self.theta_S_Ec)/2. - (np.sqrt(3)*np.cos(self.phi_S_Ec - t*omega_E)*np.sin(self.theta_S_Ec))/2.)*(np.cos(self.theta_S_Ec)*np.cos(self.theta_L) + np.cos(self.phi_S_Ec - self.phi_L)*np.sin(self.theta_S_E
np.sin(2*(t*omega_E + ArcTan(-2*np.sin(self.theta_S_Ec)*np.sin(self.phi_S_Ec - t*omega_E),np.sqrt(3)*np.cos(self.theta_S_Ec) + np.cos(self.phi_S_Ec - t*omega_E)*np.sin(self.theta_S_Ec)))))**2)/4.)) -
(np.sqrt(3)*np.sqrt(self.a0**2*(np.cos(self.theta_S_Ec)*np.cos(self.theta_L) + np.cos(self.phi_S_Ec - self.phi_L)*np.sin(self.theta_S_Ec)*np.sin(self.theta_L))**2*
((np.cos(2*(t*omega_E + ArcTan(-2*np.sin(self.theta_S_Ec)*np.sin(self.phi_S_Ec - t*omega_E),np.sqrt(3)*np.cos(self.theta_S_Ec) + np.cos(self.phi_S_Ec - t*omega_E)*np.sin(self.theta_S_Ec))))*
(1 + (np.cos(self.theta_S_Ec)/2. - (np.sqrt(3)*np.cos(self.phi_S_Ec - t*omega_E)*np.sin(self.theta_S_Ec))/2.)**2)*
np.sin(2*ArcTan(-0.5*(np.sin(self.theta_S_Ec)*np.sin(self.theta_L)*np.sin(self.phi_S_Ec - self.phi_L)) -
(np.sqrt(3)*np.cos(t*omega_E)*(np.cos(self.theta_L)*np.sin(self.theta_S_Ec)*np.sin(self.phi_S_Ec) - np.cos(self.theta_S_Ec)*np.sin(self.theta_L)*np.sin(self.phi_L)))/2. -
(np.sqrt(3)*(-(np.cos(self.theta_L)*np.cos(self.phi_S_Ec)*np.sin(self.theta_S_Ec)) + np.cos(self.theta_S_Ec)*np.cos(self.phi_L)*np.sin(self.theta_L))*np.sin(t*omega_E))/2.,
np.cos(self.theta_L)/2. - (np.sqrt(3)*np.cos(self.phi_L - t*omega_E)*np.sin(self.theta_L))/2. -
(np.cos(self.theta_S_Ec)/2. - (np.sqrt(3)*np.cos(self.phi_S_Ec - t*omega_E)*np.sin(self.theta_S_Ec))/2.)*(np.cos(self.theta_S_Ec)*np.cos(self.theta_L) + np.cos(self.phi_S_Ec - self.phi_L)*np.sin(self.theta_S_E
np.cos(2*ArcTan(-0.5*(np.sin(self.theta_S_Ec)*np.sin(self.theta_L)*np.sin(self.phi_S_Ec - self.phi_L)) -
(np.sqrt(3)*np.cos(t*omega_E)*(np.cos(self.theta_L)*np.sin(self.theta_S_Ec)*np.sin(self.phi_S_Ec) - np.cos(self.theta_S_Ec)*np.sin(self.theta_L)*np.sin(self.phi_L)))/2. -
```

Note that the analytic derivative does exist, but it is a) too long for pretty code – you see a snippet of the 657 lines of terms for $\frac{\partial h_\alpha}{\partial \theta_S}(t)$ in python code, see delh_delthetaS.txt on Git and…

<table>
<tr><th></th><th>Numerical derivative</th><th>Analytic derivative</th></tr>
<tr><td>Shape of the Amplitude</td><td>
</td><td>
</td></tr>
<tr><td>Shape of the frequency (1 hour after 0.3*yr)</td><td>
</td><td>
</td></tr>
</table>

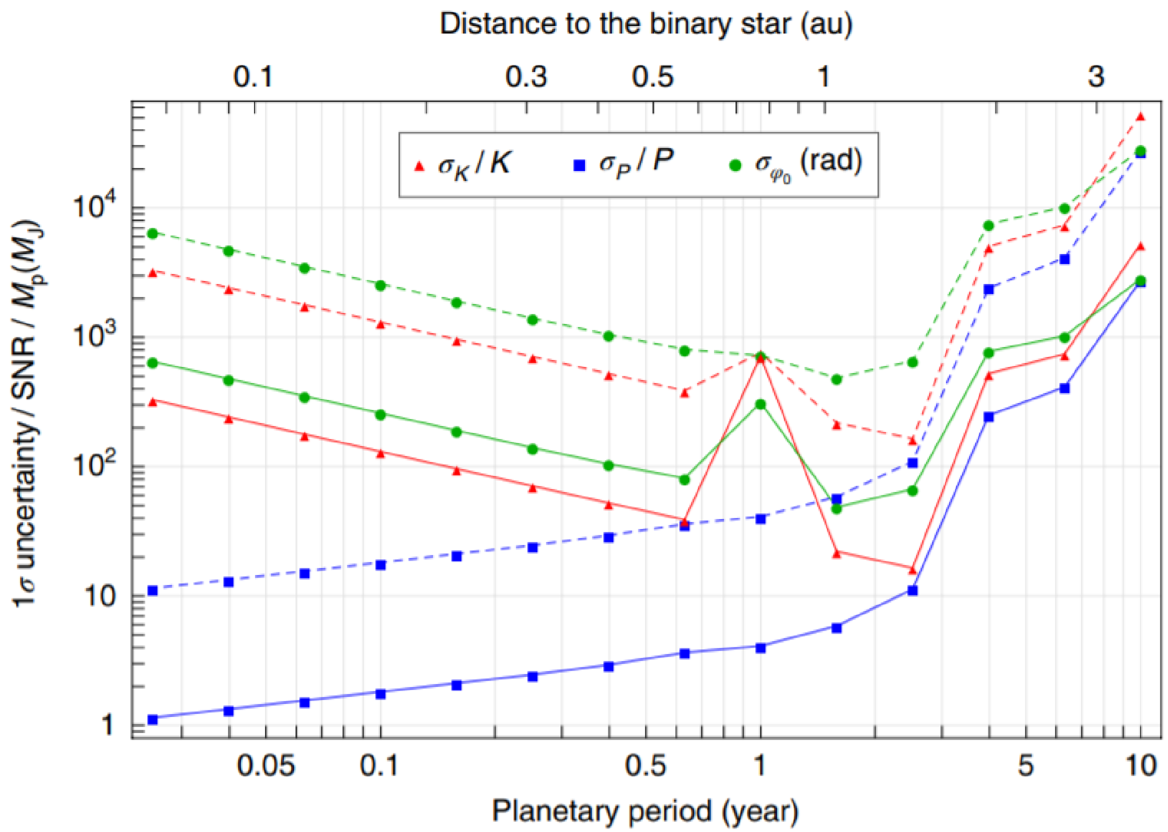
b) results in the same function anyways, though the analytic one takes longer to compute, plus…

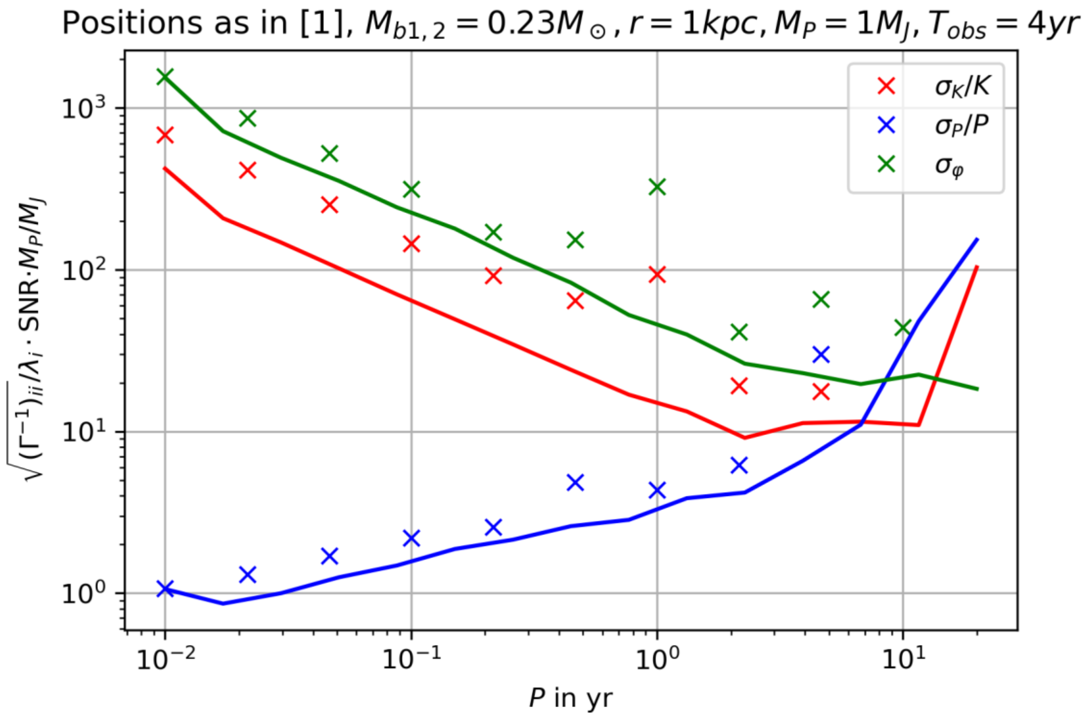
And sadly life would be too easy if we could integrate it analytically :c



c) neither `Simplify` nor `Integrate` will return

3. Result for LISA



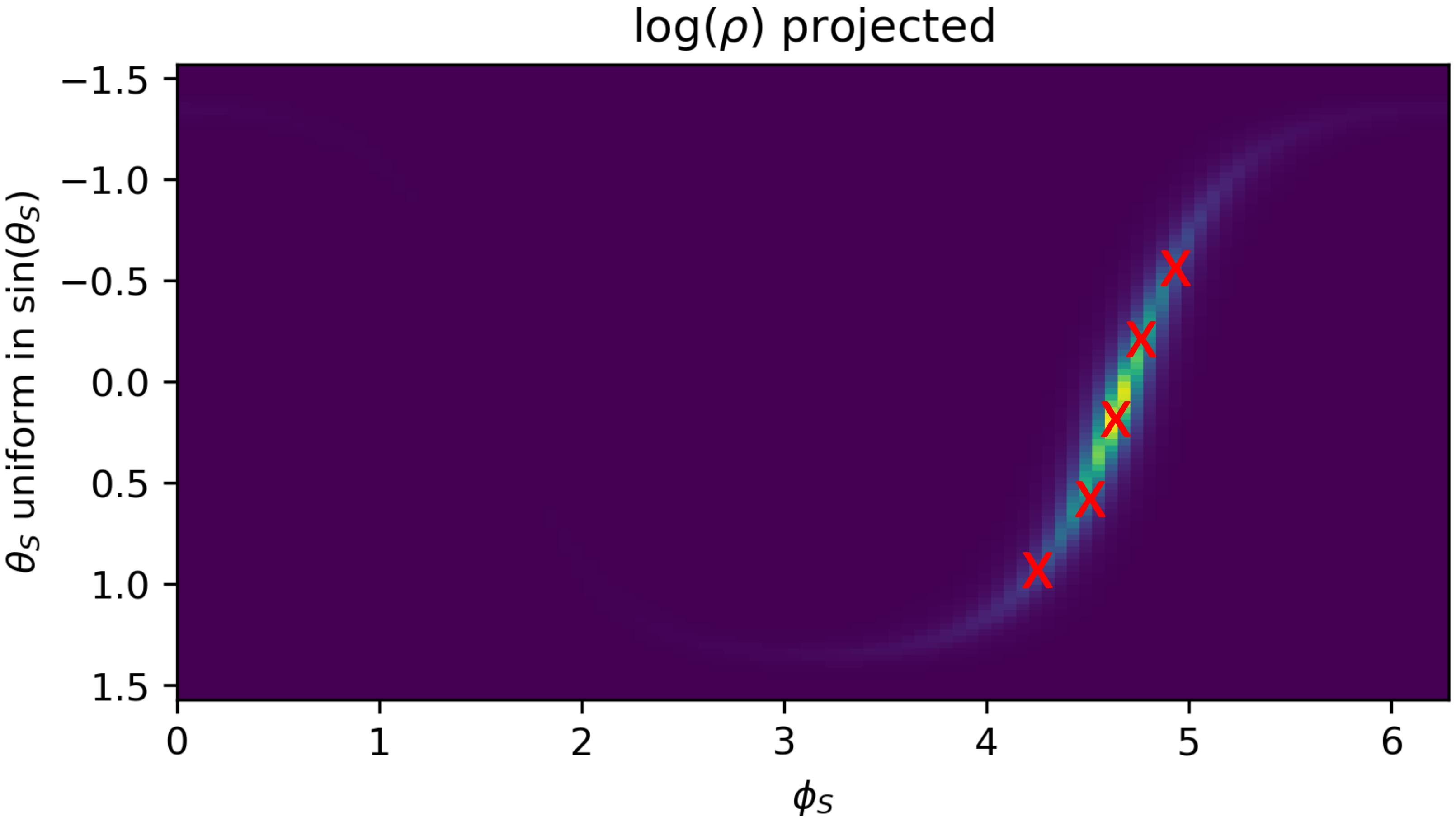


Sadly it still doesn‘t work, even with higher precision in integration… crosses are big 9x9 Fisher matrices, straight lines small 3x3 Fisher matrices

At this point I‘m finally going to stop though – only difference to Tamanini: At some point I stopped doing the integration with all three arms, but did it only with two arms and rescaled the Fisher matrix and SNR^2 by 2

How will the position be a factor for the uncertainty?

Look for a change in uncertainties over relevant region in milky way



4. Regards on exoplanet constraints in Milky Way

c.f. [2]:

DETECTIONS (4 years)

	A) $\mathcal{U}_a$ (0.1-200 au)		B) log $\mathcal{U}_a$ (0.1-200 au)		C) logNormal$_a$ (0.1-200 au)		D) $a^{-0.61}$ (0.1-200 au)	
	CBPs	BDs	CBPs	BDs	CBPs	BDs	CBPs	BDs
1) $\mathcal{U}_M$ (1M$_\oplus$ - 0.08 M$_\odot$)	**3 (0.011%)**	79 (0.302%)	**83 (0.317%)**	**2218 (8.482%)**	18 (0.069%)	503 (1.924%)	28 (0.107%)	820 (3.136%)
2) $M^{-1.31}$	6 (0.023%)	**14 (0.054%)**	30 (0.115%)	316 (1.209%)	5 (0.019%)	85 (0.325%)	13 (0.050%)	131 (0.501%)

How did they do it? Simulation:

1. Generate primary from Kroupa imf, range [.95, 10]× $M_\odot$
2. Generate partner from uniform mass ratio [0,1] and log-flat seperation, thermal eccentricity distribution
3. Place binaries in MW according to star formation rate times density
4. Add planet with $f_{\mathrm{CBP}} = 0.5$ and mass/separation distribution as in table

5. IceGiant mission

$$\mathrm{y}(t) = \frac{\mu - 1}{2}\Psi(t) - \mu\Psi\left(t - \frac{\mu + 1}{2}T_2\right) + \frac{\mu + 1}{2}\Psi(t - T_2)$$

With $\Psi(t) = \frac{\hat{n}\cdot\boldsymbol{h}(t)\cdot\hat{n}}{1-\mu^2} = \frac{h_{\hat{n}\hat{n}}(t)}{1-\mu^2}$

<u>Question</u>: What is $\hat{n} \cdot \boldsymbol{h}(t) \cdot \hat{n}$?

How well will icegiant constrain exoplanets?

Following the derivation of the pattern function in [6] §9.2.3 with setting our detector frame $(\hat{n}, \hat{y}, \hat{z})$ and the source frame $(\hat{x}', \hat{y}', \hat{k}')$ we can transform the GW tensor into the detector frame:

$$h'_{ij} = \begin{pmatrix} h_+ & h_\times & 0 \\ h_\times & -h_+ & 0 \\ 0 & 0 & 0 \end{pmatrix} \mapsto h_{ij} = R_{ik} R_{jl} h'_{kl}$$

Via a rotation by θ around $\hat{y}$, followed by φ around $\hat{z}$, which gives us R, we can then compute

$$h_{\hat{n}\hat{n}} = h_+ (\cos^2\theta \cos^2\varphi - \sin^2\varphi) + h_\times (2\cos\theta \sin\varphi \cos\varphi)$$

to find the beam pattern functions $F_+(\hat{n}, \psi = 0)$ and $F_\times(\hat{n}, \psi = 0)$

How well will icegiant constrain exoplanets?

$$F_+(\hat{n}, \psi = 0) = \cos^2\theta\cos^2\varphi - \sin^2\varphi,$$
$$F_\times(\hat{n}, \psi = 0) = 2\cos\theta\sin\varphi\cos\varphi$$

Rotation in transverse plane of ψ yields well-known transformation law of the beam pattern function and can be tuned, such that incoming wave is of the form $\boldsymbol{h}'(t) = A_+\mathbf{e}_+\cos\omega t + A_\times\mathbf{e}_\times\sin\omega t$, for this we need the polarization angle $\psi_S = \arctan\left(\frac{\hat{z}\cdot\hat{q}}{\hat{z}\cdot\hat{p}}\right)$, see [3]

-> we can change to amplitude-phase-form:

$$\Psi(t) = \frac{1}{1-\mu^2} A\cos\left(\int_0^t \omega\, dt' + \phi_p\right)$$

With $A = \sqrt{A_+^2F_+^2 + A_\times^2F_\times^2}$ and $\phi_p = \arctan\left(\frac{-A_\times F_\times}{A_+F_+}\right)$, c.f. [3]

How well will icegiant constrain exoplanets?

<u>Question</u>: What is μ? Answer: $\mu = \hat{k} \cdot \hat{n} = (\boldsymbol{R}\hat{z}) \cdot \hat{x} = \cos\varphi \sin\theta$

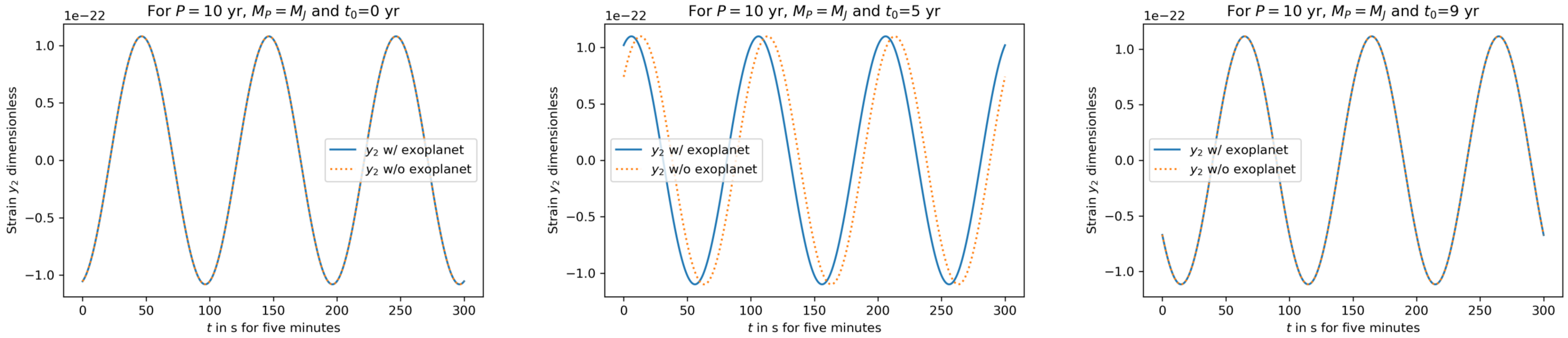
Now we have full parameter dependence of the time series:

$$\mathrm{y}(t) = \frac{\mu - 1}{2}\Psi(t) - \mu\Psi\left(t - \frac{\mu + 1}{2}T_2\right) + \frac{\mu + 1}{2}\Psi(t - T_2)$$

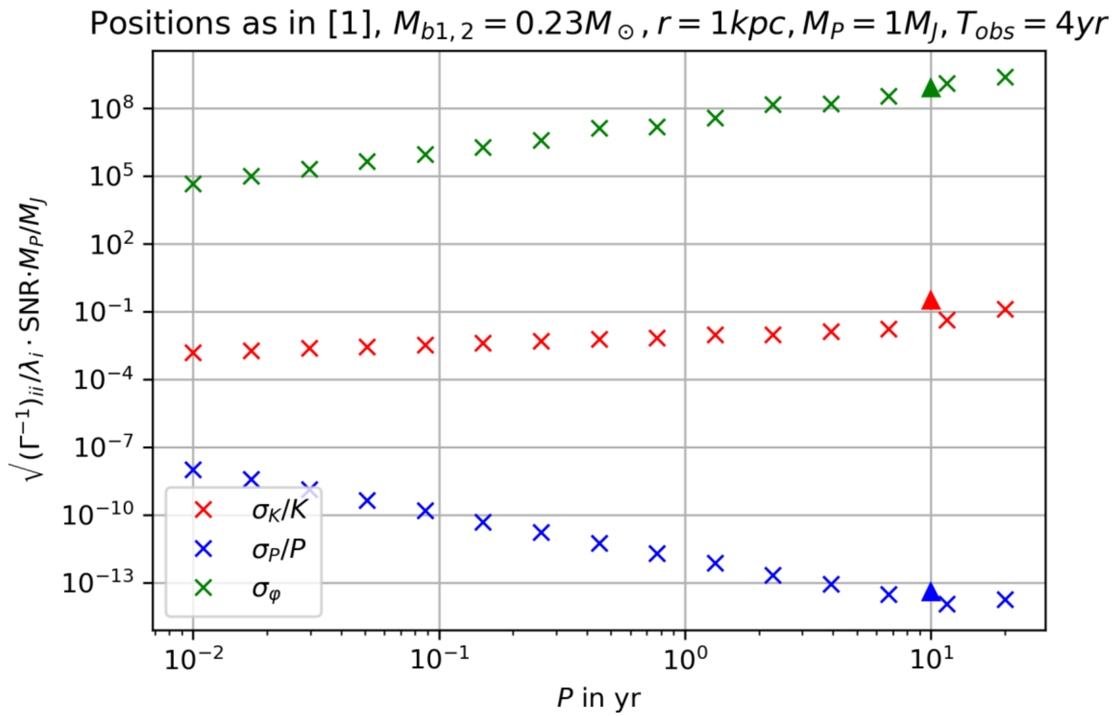
and we can perform the same Fisher information approach

But would icegiant see Jupiters?

Yes! Dependant on the SNR and position (μ= 0.27, distance 1 kpc) , but Jupiters would leave a characteristic frequency shift in the icegiant mission, about 10 s after five years if edge-on:

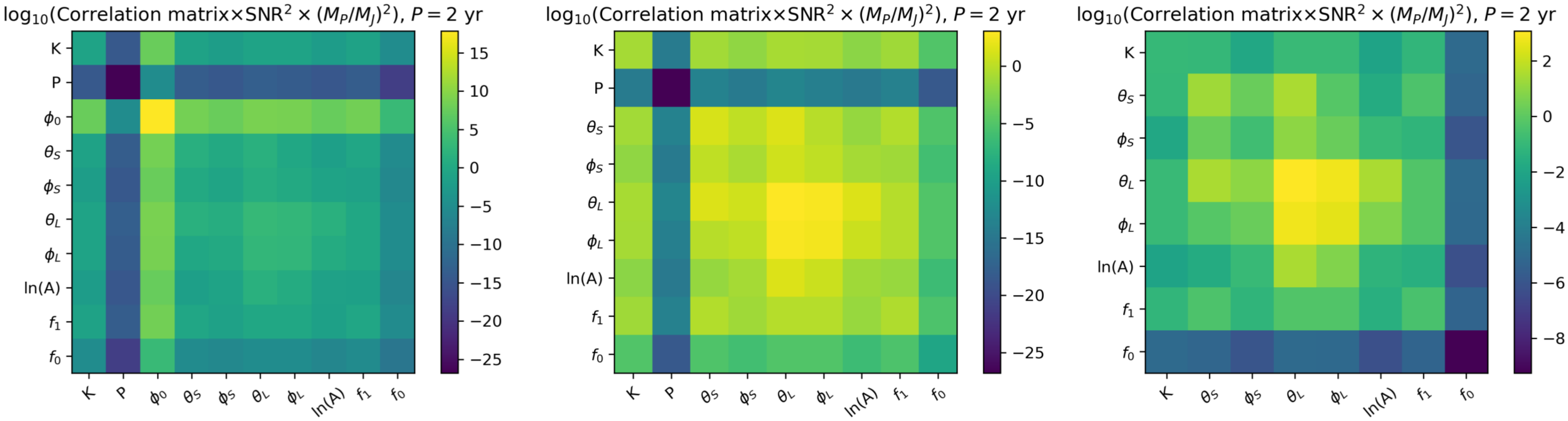


How well will icegiant constrain exoplanets?



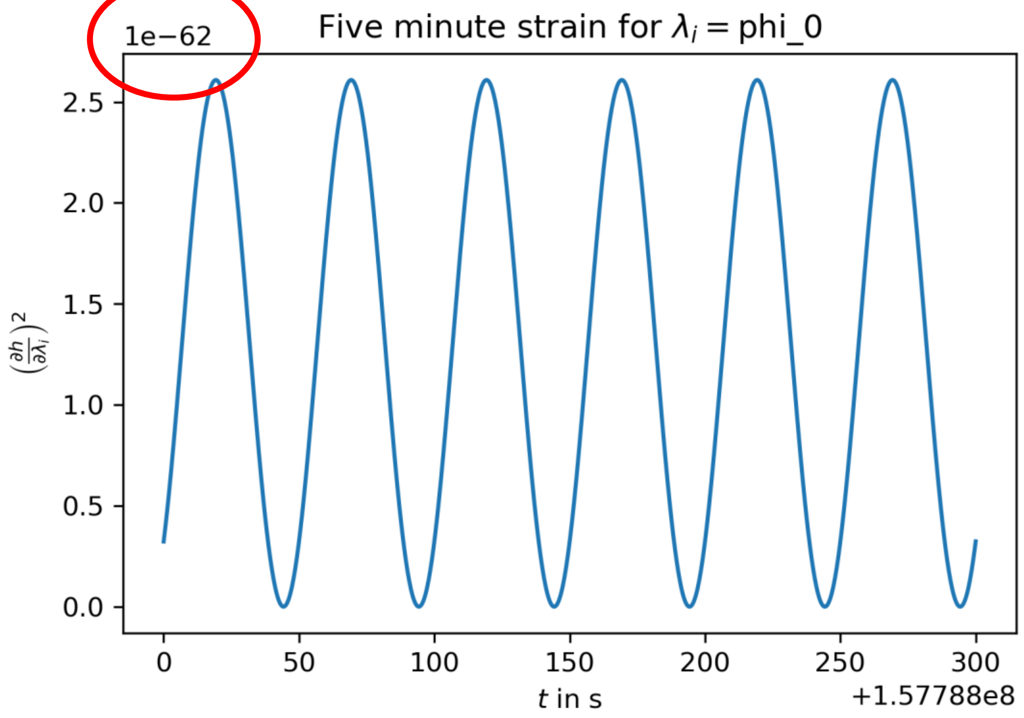
First plot using same positions as before (μ= 0.27) and only doing the small 3x3 Fisher matrix approach, i.e. ignorant result wrt. Correlations of the other parameters (triangles for full parameter approach). I used T_2 = 30000 s and 40 day measurements each year over 10 years. If we trust this plot, icegiant will find any planets with $M\sin(i) \gtrsim 1$, given LISA tells us for which waveform to look for and SNR>1 -> actual Jupiters potentially visible!

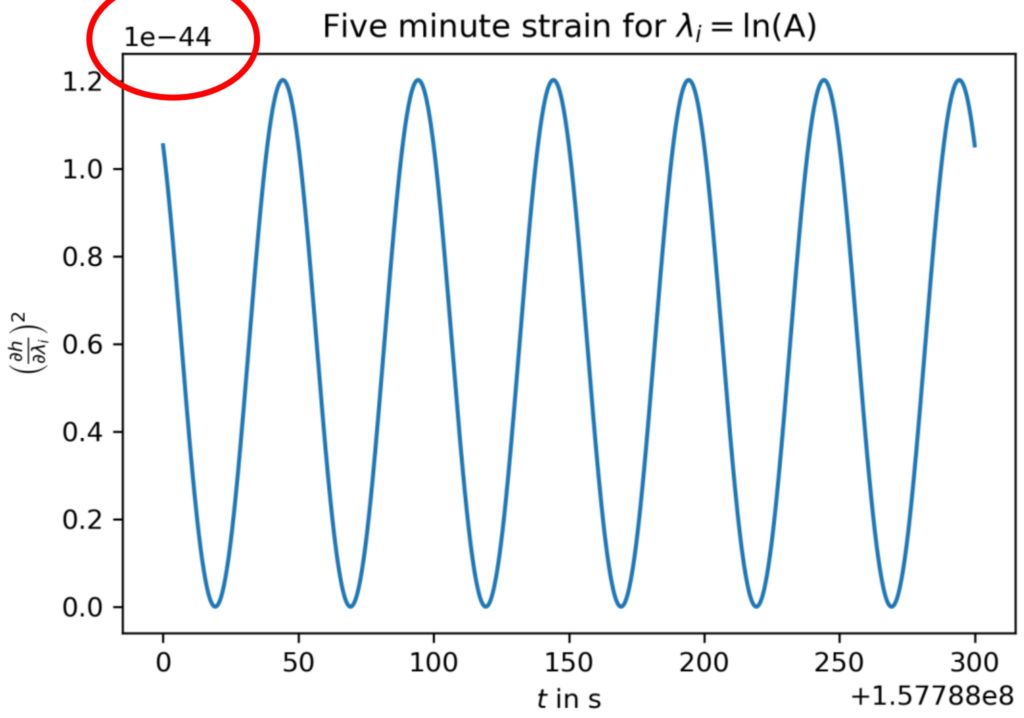
Correlation matrix for the whole parameter set



As you can see, the initial phase of the binary is completely undetermined in icegiant – qualitative explanation: only one arm measuring the change in amplitude: An exoplanet of P=10 yr coming around a f~10.1 mHz binary will produce the same strain as a P=10 yr binary going away from us at f~9.9 mHz

Correlation matrix for the whole parameter set





Alternative quantitative explanation: $\frac{\partial f}{\partial \varphi_0} = f(t)\frac{K}{c} sin\left(\frac{2\pi t}{P} + \varphi_0\right) \sim O(10^{-3} \times 10^{-7})$, so $\frac{\partial h}{\partial \varphi_0} \sim O(10^{-10} \times h)$ and therefore $\left(\frac{\partial h}{\partial \varphi_0}\frac{\partial h}{\partial \varphi_0}\right)^{-1} \sim O(10^{20} \times h^2)$ -> undetermined φ_0

Short proof for the linearity of the exoplanet relative uncertainties in K^{-1}

The strain goes always like $h(t) = A(t)\cos\left[2\pi\int_0^t f(t')dt' + \Phi(t)\right]$
with derivative $\frac{\partial h}{\partial\lambda}(t) = -A(t)\left[2\pi\int_0^t \frac{\partial f}{\partial\lambda}(t')dt' + \frac{\partial\Phi}{\partial\lambda}(t)\right]$

$$\times\sin\left[2\pi\int_0^t f(t')dt' + \Phi(t)\right]$$

In the case of $\lambda = \mathrm{P}, \varphi_0$ we have $\frac{\partial f}{\partial\lambda} \propto K$ and in the other derivatives K enters only in higher orders, so we have a block matrix in the basis $\lambda = (\mathrm{P}, \varphi_0), (K), (\dots)$: $\Gamma_{ij} = \begin{pmatrix} \propto K^2 & \propto K \\ \propto K & \propto 1 \end{pmatrix}$

Short proof for the linearity of the exoplanet relative uncertainties in K^{-1}

$\Gamma_{ij} = \begin{pmatrix} \propto K^2 & \propto K \\ \propto K & \propto 1 \end{pmatrix}$ with $\det(\Gamma) \propto K^4$ and with $\Gamma^{-1} = \frac{1}{\det(\Gamma)} \operatorname{adj}(\Gamma)$

We find $\Gamma^{-1} = \frac{1}{\det(\Gamma)} \operatorname{adj}(\Gamma) \propto K^{-4} \begin{pmatrix} \propto K^2 & \propto K^3 \\ \propto K^3 & \propto K^4 \end{pmatrix}$, so

$\Gamma^{-1} \propto \begin{pmatrix} \propto K^{-2} & \propto K^{-1} \\ \propto K^{-1} & \propto 1 \end{pmatrix}$ and then we want the correlation matrix from the covariance matrix, so in the basis as before $\lambda = (\mathrm{P}, \varphi_0), (K), (\dots)$

$$\operatorname{corr}(i,j) = \frac{\Gamma^{-1}}{\lambda_i \lambda_j} \propto \begin{pmatrix} \propto K^{-2} & \propto K^{-2} & \propto K^{-1} \\ \propto K^{-2} & \propto K^{-2} & \propto K^{-1} \\ \propto K^{-1} & \propto K^{-1} & 1 \end{pmatrix}$$

Combining measurements

The big advantage for Fisher matrix approach: Adding measurements is fairly easy: We have for $k = L$ for LISA and $k = IG$ for IceGiant

$$\Gamma_{ij}^{(k)} = \frac{2}{S_n^{(k)}(f_0)} \int_0^{T_0} dt \, \frac{\partial h_\alpha}{\partial \lambda_i}(t) \, \frac{\partial h_\alpha}{\partial \lambda_j}(t)$$

And for the combined measurement $\Gamma_{ij}^{(tot)} = \Gamma_{ij}^{(L)} + \Gamma_{ij}^{(IG)}$ which we can rescale by dividing the total signal to noise ratio $SNR^2 = SNR_{(L)}^2 + SNR_{(IG)}^2$, with

$$SNR_{(k)}^2 = \frac{2}{S_n^{(k)}(f_0)} \int_0^{T_0} dt \, h_\alpha(t) h_\alpha(t)$$

Combining measurements

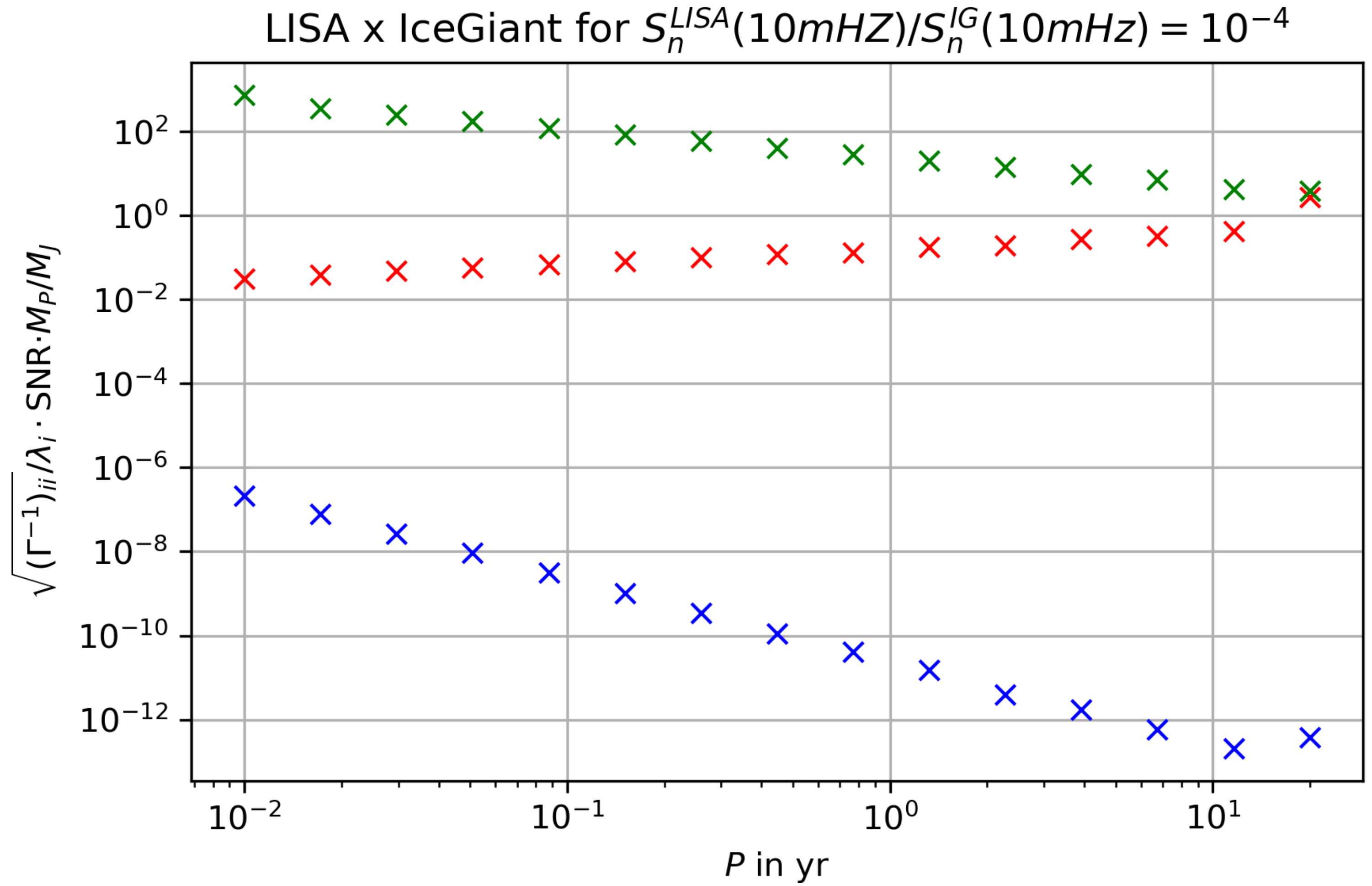
By cleverly defining the weight $w = \frac{S_n^{(L)}(f_0)}{S_n^{(IG)}(f_0)} < 1$ we find

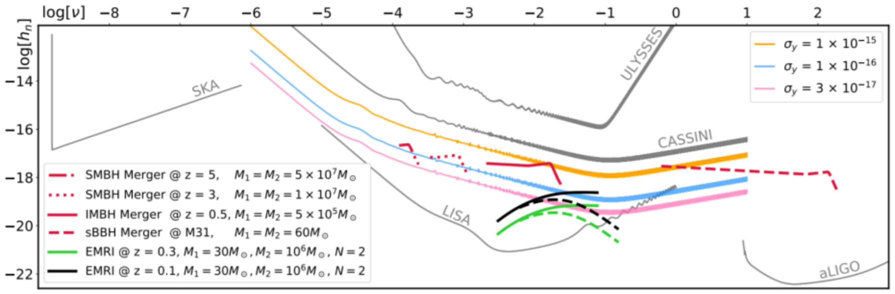
$SNR^2 = SNR_{(L)}^2 + SNR_{(IG)}^2 = SNR_{(L)}^2 \left(1 + w \frac{\int y^2 dt}{\int h^2 dt}\right)$ and in total

$$\frac{\Gamma_{ij}^{(tot)}}{SNR^2} = \frac{\Gamma_{ij}^{(L)}}{SNR_{(L)}^2}\left(1 + w \frac{\int y^2 dt}{\int h^2 dt}\right)^{-1} + w \frac{\int \frac{\partial y}{\partial \lambda_i}\frac{\partial y}{\partial \lambda_j} dt}{\int h^2 dt + \int y^2 dt}$$

Note that now w is a free parameter, dependant on the strain sensitivity of an icegiant mission in comparison to LISA's strain sensitivity at a given frequency

First combination





I got the 1e-4 from the plot above from the IceGiant paper – given I understood it correctly: Is it log10? Is $h_n(f_0) = S_n^{(IG)}(f_0)$?
As I already did the integrations and saved them, you can also play around with it: It’s comparison(mode=‘s’,weight=w) in LISA_x_IceGiant.py – mode=‘m’ (9x9 Fisher matrix) is in preperation

Next milestones

✓Understand the fluctuations a bit better and potentially fix them

✓Do the calculations in Mathematica

✓Add $\ln(A)$, f_1 fit to see actually the same plot as Tamanini

✓Look at the parameter space for positions/angular momentum for the 25‘000 potential DWDs with SNR > 7 and calculate positional dependance of the planetary parameters -> we want a function:

```
rel_uncertainty(pos, M, sep)
```

-> Unrealistic to compute a good grid in limited time

✓Assuming a prior on the DWD parameters and planetary parameters (mass, inclination, separation), we can then take the integral as in [4], constraining f_{CBP} the fraction of circumbinary partners given N_{bin} detections via Bayesian inference

Next milestones

$$N_{\text{bin}} = \int_0^z \int_0^\infty \int_{\nu_{\text{ISCO}}}^{\nu} \frac{\mathrm{d}V}{\mathrm{d}z} \frac{\mathrm{d}\mathcal{R}}{\mathrm{d}M} \mathcal{D}(M, q, \nu, z) \frac{\mathrm{d}\nu}{\dot{\nu}_{\text{GW}}} \mathrm{d}M \mathrm{d}z, \quad (11)$$

$$\text{with} \quad \mathcal{D}(M, q, \nu, z) \equiv \mathcal{H}\left[h_c(M, q, \nu, z) - \rho_c h_n(\nu)\right], \quad (12)$$

-> already done by Danielsky et al. [2]

✓ Then repeat the exercise with the strain and signal to noise of the IceGiant mission:

$$y_2^{\text{GW}}(t) = \frac{\mu - 1}{2}\bar{\Psi}(t) - \mu\bar{\Psi}\left(t - \frac{\mu + 1}{2}T_2\right) + \frac{\mu + 1}{2}\bar{\Psi}(t - T_2), \quad (1)$$

to see if it could see most promising exoplanet candidates/Jupiter-like planets

✓ Combine measurements of IceGiant and LISA -> just add Fisher information prior to inversion ☺

- Look for extragalactic X-Ray binaries and constrain the number of exoplanets/stars in orbit around them

References

[1] Tamanini, N., Danielski, C. (2019). The gravitational-wave detection of exoplanets orbiting white dwarf binaries using LISA. *Nat Astron 3*, 858–866. https://doi.org/10.1038/s41550-019-0807-y

[2] Danielski, C., Korol, V., Tamanini, N., & Rossi, E.M. (2019). Circumbinary exoplanets and brown dwarfs with the Laser Interferometer Space Antenna. *Astronomy and Astrophysics, 632.*

[3] Cutler, C. (1998). Angular resolution of the LISA gravitational wave detector. *Physical Review D, 57*, 7089-7102.

[4] Soyuer, D., Zwick, L., D'Orazio, D., Saha, P. (2021). Searching for gravitational waves via Doppler tracking by future missions to Uranus and Neptune. *MNRAS: Letters*, 503, 1, L73-79. https://doi.org/10.1093/mnrasl/slab025

[5] Maggiore, M. (2008). *Gravitational Waves Volume 1: Theory and Experiments*. Oxford University Press