Problem Set 5: The cycle of twelve

Sand mouse cells have a circadian oscillator that partly depends on transcriptional regulation. The mRNA expression levels of twelve sand mouse genes are known to vary cyclically in a 24 hour period, in different phases. However, their phases relative to each other remain unknown. The phase information is critical. If the 12 phases were known, then the 12 genes could be ordered into a relative order of activation. This would make testable predictions about which gene products are most likely to be 'upstream' in the regulatory pathway and directly regulating transcription of which 'downstream' genes.

Available data

Of course, you could do RNA-seq experiments at different time points in entrained (synchronized) sand mouse cells, but it would take time and money, and you and your lab mates are eager to leap ahead quickly to mechanism. By searching the literature and the GEO database, you've found that there are already a few published RNA-seq experiments in entrained sand mouse cells, at a small number of time points. You've collated these data into a table (`pset5-data.tbl`):

gene	4hr, +-20	4hr, +-2	8hr, +-20	8hr, +-5	16hr, +-5	16hr, +-20	24hr, +-2	24hr, +-20
anise	23.76	23.19	33.33	29.36	58.09	53.23	32.63	23.48
kiwi	74.46	54.53	59.97	24.63	34.45	0.00	62.12	25.04
carrot	62.08	72.83	31.45	49.08	12.61	18.49	76.04	91.07
grape	55.74	43.46	55.83	65.21	40.75	57.33	18.13	29.24
tangerine	0.00	18.83	28.75	39.69	65.80	45.29	17.48	61.24
melon	48.00	27.93	53.07	57.49	60.84	57.65	17.38	3.83
clementine	72.22	56.16	49.29	72.70	30.23	42.15	32.11	0.00
spinach	39.62	27.26	0.00	33.72	59.37	70.99	47.72	29.73
beet	61.48	43.18	15.43	28.25	37.14	43.53	61.96	40.37
huckleberry	36.77	31.89	0.00	20.23	51.81	87.98	53.55	66.11
lentil	53.58	68.50	83.86	78.84	15.83	9.53	43.90	0.00
cauliflower	49.54	65.97	77.94	54.80	9.68	30.73	54.75	20.60

There are a total of 8 experiments, with two experiments each at time points at 4, 8, 16, and 24 hours, as marked in the first line of bold column headers. The numbers in the table are estimated expression levels in TPM for the 12 genes.

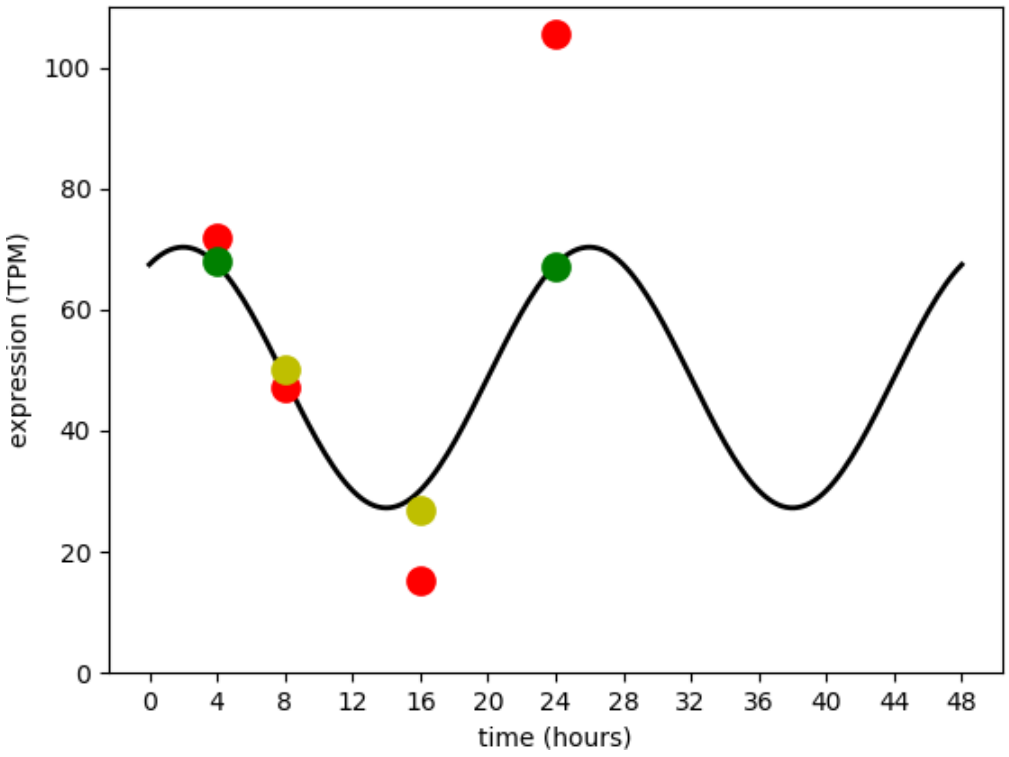

Figure 1. An example of a sand mouse gene being expressed in a circadian rhythm, with a period of 24 hours. The true average oscillation in expression level is shown with the black line. The eight available experiments have measured this gene's expression at 4, 8, 16, and 24 hour time points. Two experiments are accurate with low variance (green dots); four experiments are inaccurate with high variance (red dots); two experiments are intermediate (yellow dots).

The catch is that the experiments have different reliabilities, as indicated in the second line of bold column headers marked +-20, +-2, and +-5. This variability is partly because different numbers of replicates were done, and partly because some experiments were more careful than others. For the purposes of the exercise, we will assume that the errors in TPM measurements are Gaussian-distributed with these known standard deviations σ. For example, the first experiment is a 4hr time point, with normal error $N(0, \sigma = 20)$. There are two high accuracy experiments with $\sigma = 2$ at 4hr and 24hr points; four low accuracy experiments ($\sigma = 20$); and two moderate accuracy experiments ($\sigma = 5$).

You can assume that the underlying model for an cyclic expression level y_t for a gene at time t (in hours) is

$$y_t = b + a \sin(2\pi\omega(t + \phi))$$

in terms of three parameters: a baseline mean expression level b (in TPM), an oscillation amplitude a (in TPM), and a phase ϕ (offset, in hours). You can treat the circadian frequency ω as known, $\frac{1}{24\text{hr}}$. Note that the term $2\pi\omega(t + \phi)$ ends up in units of radians (That's a hint. Don't confuse hours and radians in implementing this exercise).

The game is to deduce the unknown phases ϕ, given this small amount of published RNA-seq data (`pset5-data.tbl`) of variable reliability, and thereby deduce the correct relative order of the 12 genes. (If mRNA1 has a ϕ +2hrs of mRNA2, that means mRNA1 starts to rise two hours before mRNA2 does, and thus is more likely to be a direct regulator of mRNA2 transcription, as opposed to the other way around.)

Moriarty's solution

Moriarty, the brilliant postdoc in the lab who's a bit full of himself, says he's seen this problem before -- it's **harmonic regression**. He's familiar with a clever and widely used trick for it. The equation above for y_t is nonlinear in ϕ (because of the $\sin$), but Moriarty remembers his angle addition formulas from trigonometry, one of which is:

$$\sin(\alpha + \beta) = \sin\alpha\cos\beta + \sin\beta\cos\alpha$$

If we apply that to the modeling equation for y_t we can get a linear regression function with three parameters:

$$y_t = p_0 + p_1 \sin(2\pi\omega t) + p_2 \cos(2\pi\omega t)$$

where those three parameters are:

$$\begin{aligned} p_0 &= b \\ p_1 &= a\cos(2\pi\omega\phi) \\ p_2 &= a\sin(2\pi\omega\phi) \end{aligned}$$

The function is nonlinear in t, but it's linear in the parameters, which is what matters for solving for ϕ. Moriarty says, once we're in this form, we can just solve for the parameters with ordinary least squares. Given a least squares fit for the parameters, $\frac{p_2}{p_1} = \tan(2\pi\omega\phi)$ and we can use $\arctan$ to solve for $\hat{\phi}$; then given $\hat{\phi}$, we can get $\hat{a}$ from either p_1 or p_2.

Moriarty's script (`moriarty.py`), takes the data table as input, and does an ordinary least squares fit for each gene's expression data to obtain best fit $\hat{b}$, $\hat{a}$, and $\hat{\phi}$. Ordering the results by $\hat{\phi}$, Moriarty deduces that the pathway order appears to be:

gene	b (tpm)	a (tpm)	phase \(\phi\) (hr)
cauliflower	40.60	25.65	23.32
lentil	38.06	42.31	22.39
clementine	41.89	27.75	21.39
grape	46.05	21.31	19.76
melon	44.01	28.86	18.22
anise	38.76	16.61	14.44
tangerine	39.52	19.77	13.42
spinach	43.27	22.72	11.75
huckleberry	48.45	34.77	10.29
beet	40.63	17.22	7.30
carrot	44.81	37.24	4.81
kiwi	36.53	21.48	2.13

But even as Moriarty goes around the lab crowing about his solution, designing incisive new experiments based on his brilliant deduction of the order (or so he says), you're not quite convinced. The uneven reliability of the eight experiments bothers you. You're not sure the implicit assumptions of least squares fitting have been met.

1. Solve by maximum likelihood

Given the RNA-seq data (`pset5-data.tbl`), use maximum likelihood to solve for the unknown phases ϕ.

For each gene independently, you'll need to calculate a log likelihood $\log P(\text{data} \mid \theta)$, where the observed data are the expression levels in the 8 experiments, and the parameters include the unknown parameters b, a, and ϕ you want to find, plus the known parameters σ for each experiment.

Then you want to identify ML estimates $\hat{a}$, $\hat{b}$, and $\hat{\phi}$ for each gene, by identifying the parameters that maximize the log likelihood.

You could discretize (as we've done before) and enumerate a grid search over a range of parameter values, but with three parameters in play, that'd get to be tedious. Instead, this week it's time to learn a new trick: solve this one using multidimensional optimization using SciPy's `optimize.minimize()` function.

The interface to calling any general library's optimization routine (SciPy or otherwise) is always finicky. You give it your objective function (here, your log likelihood calculation) in a general form, so SciPy can call it blindly and doesn't need to worry about what anything means. All it needs to do is start from an initial guess at a parameter vector $\bar{p}$ (provided by you) and move $\bar{p}$ around in parameter space, calling the objective function you provided every time it moves to a new point, until it finds a point $\bar{p}$ such that your objective function $f(\bar{p})$ reaches an optimum. There's a ton of art in making optimizers work properly, but since that's been done for you in SciPy, essentially all the art (from your point of view) is in figuring out the interface to optimize.minimize(). Documentation for it is here, and see the hints at the end for an example.

2. Compare solutions

Moriarty is incensed that you have a different solution. He offers to bet you on who's right. Compute the total log likelihood (summed over all 12 genes) for Moriarty's solution and yours. Which is more likely, and by how much?

3. Plot the fits

For each of the 12 genes, plot the observed data and the two fits (so you can show Moriarty how he went wrong).

(You can use something like a 3x4 multipanel figure in `matplotlib`.)

Hint

`optimize_example.py` is a heavily commented example of using `scipy.optimize.minimize()` on a simple maximum likelihood optimization.