Bayesian modeling of ecological systems using the ‘Stan’ software package

Greg Britten & J. Paul Mattern

January 16, 2020

Python code & Stan `functions` block

Python:

```python
import numpy as np
def npz(x,t,theta):
    '''
    input
    =====
    x: model state
    t: current time (in days)
    theta: model parameters
    '''
    n = max(0.0, x[0])
    p = max(0.0, x[1])
    z = max(0.0, x[2])

    light = 1.0 + 0.5*(theta['irr']*np.sin(
            np.pi*((t-81.25)/182.5))-theta['irr'])
    growth = theta['vmax']*n/(theta['nuthalfsat']+n)*light*p
    grazing = theta['graz'] * p*z
    ploss = theta['mort_p'] * p
    zloss = theta['mort_z'] * z*z
    return np.array((-growth+ploss+zloss,
                     growth-grazing-ploss,
                     grazing-zloss))
```

Stan:

```stan
functions {
    real[] npz(real t,       // time
               real[] x,     // state
               real[] theta, // parameters
               real[] x_r,   // fixed real data (empty)
               int[] x_i) {  // fixed integer data (empty)
    /* theta[1]: vmax, theta[2]: nuthalfsat, theta[3]: graz,
    theta[4]: mort_p, theta[5]: mort_z, theta[6]: irr */

    real n = fmax(0.0, x[1]);
    real p = fmax(0.0, x[2]);
    real z = fmax(0.0, x[3]);

    real light = 1.0 + 0.5*(theta[6]*sin(
                 pi()*((t-81.25)/182.5))-theta[6]);
    real growth = theta[1]*n/(theta[2]+n) * light * p;
    real grazing = theta[3] * p*z;
    real ploss = theta[4] * p;
    real zloss = theta[5] * z*z;
    return {-growth+ploss+zloss,
            growth-grazing-ploss,
            grazing-zloss};
    }
}
```