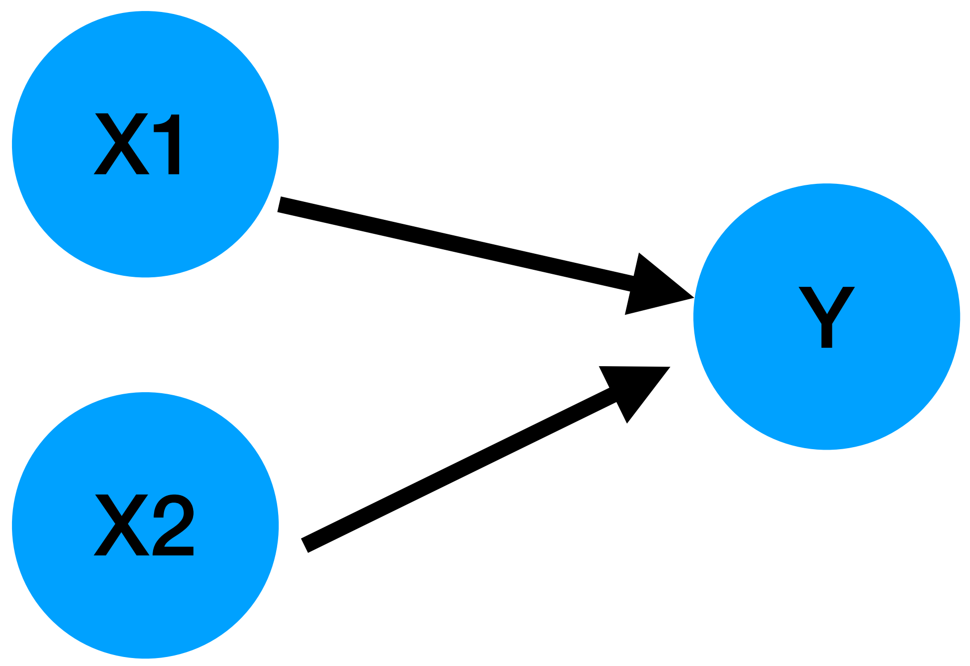
From Boolean states to ODEs using logic approximations



Boolean networks representation

$$Y(t+1) = f_b(X_1(t), X_2(t), \ldots)$$

Logic approximation representation

$$\frac{dY}{dt} = \beta\Theta(f_b(X_1, X_2, \ldots)) - \alpha Y$$

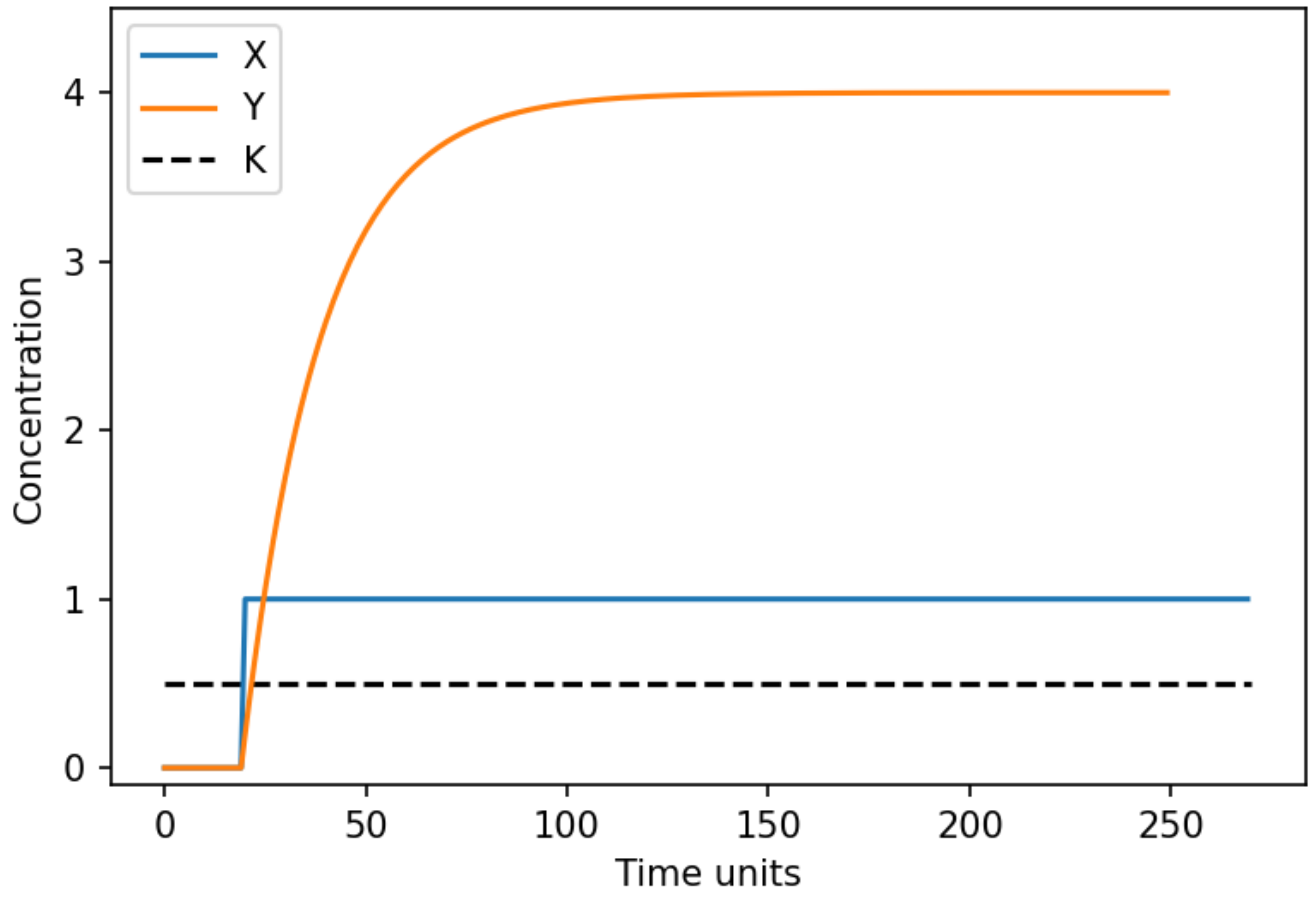
$$\Theta(f_b) = \begin{cases} 0, & \text{if } f_b = \text{True}; \\ 1, & \text{otherwise.} \end{cases}$$

Logic approximation, example

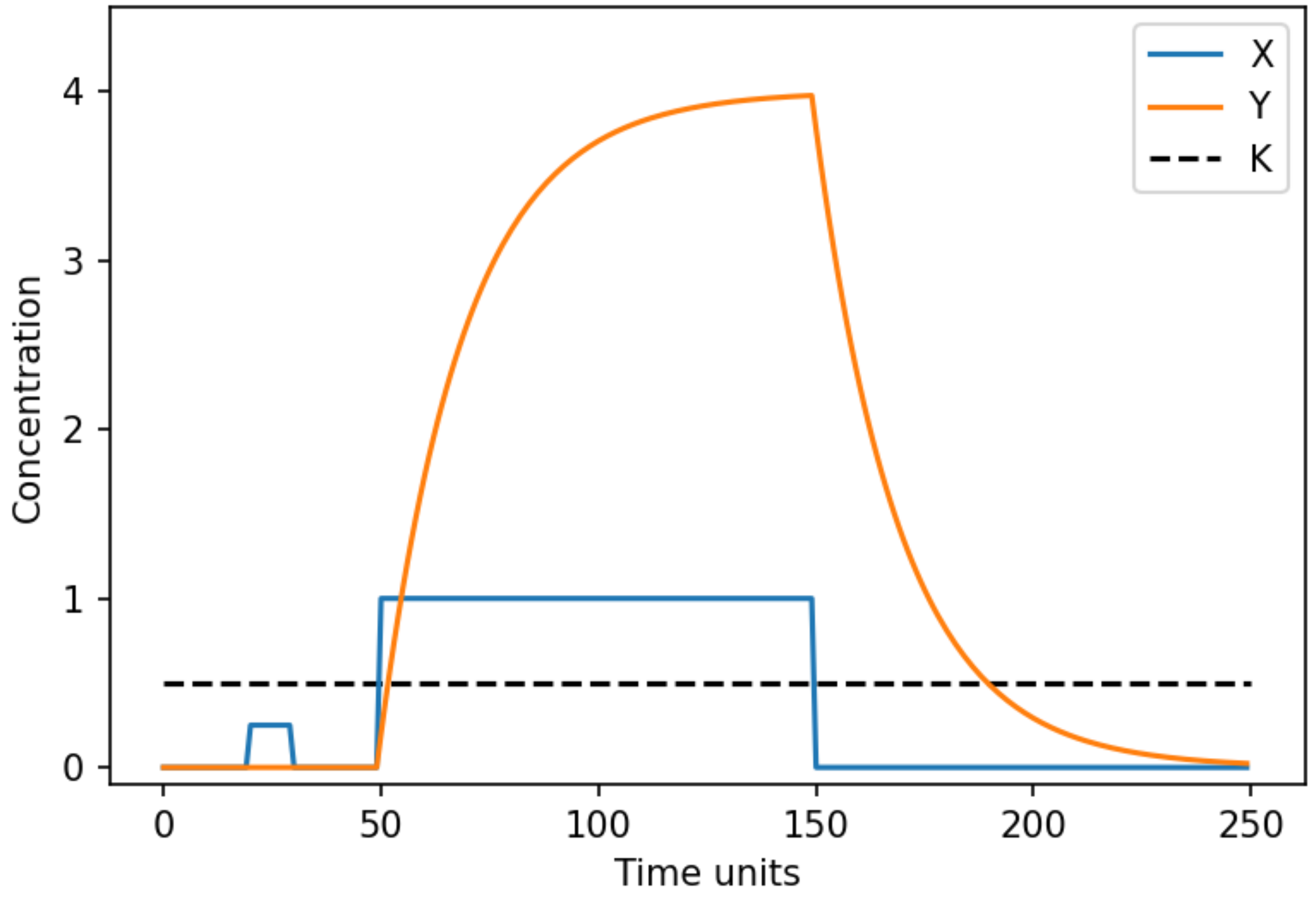
Simple activation

$$\frac{dY}{dt} = \beta\Theta(X_1 > K_x) - \alpha Y$$

Constant input, X



Pulsed input, X



Let's implement this in Python

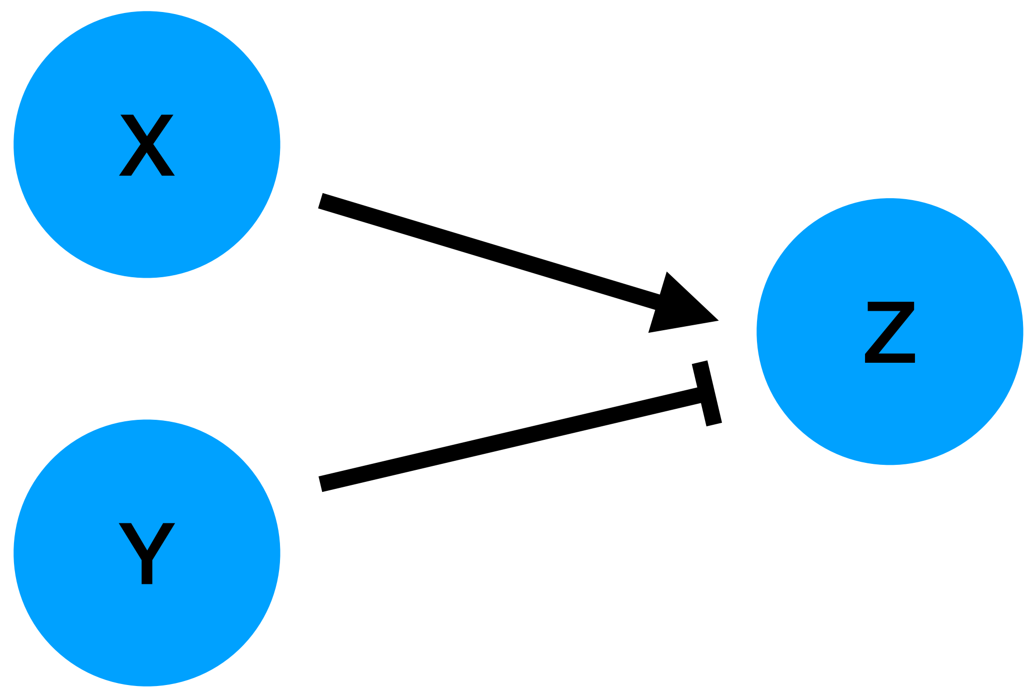
Simple activation

$$\frac{dY}{dt} = \beta\Theta(X_1 > K_x) - \alpha Y$$

Logic approximation, example

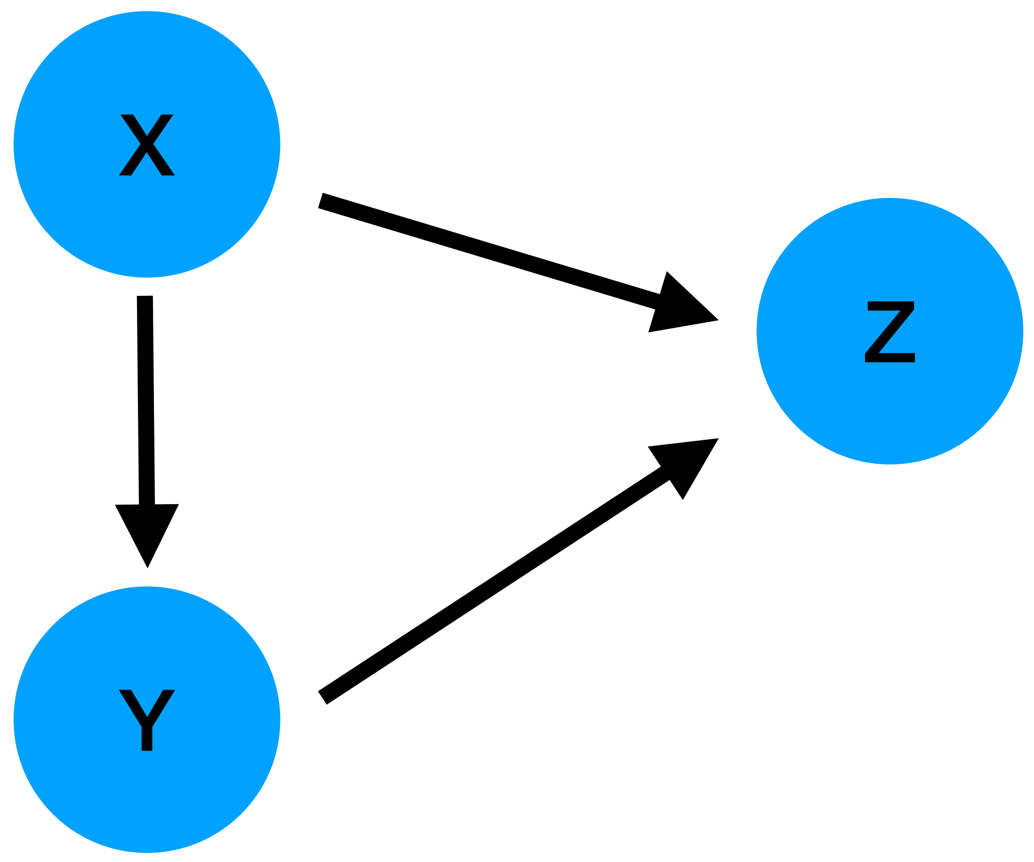
Two inputs

$$\frac{dZ}{dt} = \beta\Theta(X > K_x \wedge Y < K_y) - \alpha Z$$



A coherent feed-forward loop (FFL)

$$\frac{dY}{dt} = \beta_y \Theta(X > K_{x,y}) - \alpha_y Y$$



$$\frac{dZ}{dt} = \beta_z \Theta(X > K_{x,z} \wedge Y > K_{y,z}) - \alpha_z Z$$