# 1 Theoretical study

# 2 Numerical study

## Omegas – normally distributed

## 2.1 $r_\infty(K)$ – simulated vs predicted

### 2.1.1 Euler method

**Design and initiation** My implementation of the Euler method is object-oriented. This may not be the computationally most efficient way, but I'm still somewhat of a beginner with code, so I appreciate the clarity that object-oriented programming affords.

The first type of object is simply an `Oscillator` with three variables: $\omega, \theta_{t-1}$ and $\theta_t$. A $\Delta t = 1$ is enough in the Euler method. An `Oscillator` gets initiated with its natural frequency and the current-period $\theta$. The last-period $\theta$ is initiated as zero, since the first part of making an 'Euler step' forwards is to hand over $\theta_t$ to $\theta_{t-1}$.

The other object type is the `OscPopulation` . It consists of a list of `Oscillators` and a construction mode, namely the distribution according to which the $\omega$s are distributed. When an `OscPopulation` is initiated, for each `Oscillator` a natural frequency and initial phase are drawn from the respective type of random distribution. An `Oscillator` object is then initiated with those values and assigned to a place in the list within the `OscPopulation` object.

**Running the simulation**

### 2.1.2 Numerical integration

The strategy to integrate the consistency equation is to loop through different values of $r$ and check which ones make the right-hand side approximately equal to one. In a loop around this one, we cycle through the different values for K to get the values of the desired $r_\infty(K)$ relationship.

## 2.2 $r(t)$ for $K = [1, 2]$

## Omegas – uniformly distributed

## 2.3 $r_\infty(K)$

## 2.4 $r(t)$ for different initial conditions $\theta_0$

## 2.5 $r(t)$ for different initial conditions $\omega_0$