The equation $H = -q(1-q)$ defines a constant of motion for the SD part of the game, where q is the fraction of players in the 4th strategy. Using the transition probabilities of the different process we can derive an expression for the expected change in H within the simplex.

Where $i, j, k, N-i-j-k$ are the players playing R, P, S, and the 4th strategy respectively.

$\Delta H = H(t+1) - H(t),$
$\Delta H = -x_{t+1}(1-x_{t+1}) - (-x_t(1-x_t))$
$\Delta H = -x_{t+1}(1-x_{t+1}) + x_t(1-x_t)$
$\Delta H = x_t(1-x_t) - x_{t+1}(1-x_{t+1})$
Rough equation:
$\langle \Delta H \rangle = \sum_{i,j,k} \Big(\Delta H_s - \Delta H_{s'}\Big) T^{s\to s'}$, s is a particular state in the simplex.

$$\begin{aligned}\langle \Delta H \rangle = \text{scaling ?} \ \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{N} \Big[&(N-i-j-k)(1-N+i+j+k)(T^{R+} + T^{P+} + T^{S+} + T^{+R} + T^{+P} + T^{+S}) \\ &- (N-i-j-k+1)(-N+i+j+k)T^{R+} \\ &- (N-i-j-k+1)(-N+i+j+k)T^{P+} \\ &- (N-i-j-k-1)(2-N+i+j+k)T^{+R} \\ &- (N-i-j-k-1)(2-N+i+j+k)T^{+P} \\ &- (N-i-j-k-1)(2-N+i+j+k)T^{+S}\Big]\end{aligned} \tag{1}$$

$q = 1-x-y-z$, $p = N-i-j-k$

$$\begin{aligned}\langle \Delta H \rangle = \text{scaling ?} \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{N} \Big[&p(1-p)(T^{R+} + T^{P+} + T^{S+} + T^{+R} + T^{+P} + T^{+S}) \\ &- (p+1)(-p)(T^{R+} + T^{P+} + T^{S+}) \\ &- (p-1)(2-p)(T^{+R} + T^{+P} + T^{+S})\Big]\end{aligned} \tag{2}$$

$$\begin{aligned}\langle \Delta H \rangle = \text{scaling?} \int_0^1 dx \int_0^1 dy \int_0^1 dz \Big[&q(1-q)(T^{R+} + T^{P+} + T^{S+} + T^{+R} + T^{+P} + T^{+S}) \\ - (q+\frac{1}{N})(1-q-\frac{1}{N})(T^{R+} + T^{P+} + T^{S+}) &- (q-\frac{1}{N})(1-q+\frac{1}{N})(T^{+R} + T^{+P} + T^{+S})\Big]\end{aligned} \tag{3}$$

Numerical integration in python code ./augRps.py, shows change of sign as expected. Matches nicely with the approximated values for the Moran process. The specific expression for Moran process is very long. Computed numerically and solved with scipy.integrate (reference scipy)