# B Supplementary Materials (not for publication)

## B.1 Lowerbound discount factor under concave preferences

It is standard in the experimental literature on indefinitely repeated games to calculate payoffs based on the assumption of homogeneous players with linear preferences. Our analysis has followed this standard approach. Here, we clarify what would happen if we explicitly included risk aversion by calculating the lowerbound discount factor under CRRA preferences. We find that for some empirically reasonable CRRA coefficients full cooperation remains an equilibrium. The discussion offered below is based on the analysis in Camera et al. (forth.).

For generality, consider the following payoff matrix:

| | Producer's choice | |
|---|---|---|
| | *Help* ($C$) | *Do not help* ($D$) |
| Payoff to producer | $b$ | $y$ |
| Payoff to consumer | $a$ | $x$ |

Let players be homogeneous and expected utility maximizers, but relax linearity of preferences assuming utility $u(c) = c^{1-\gamma}/(1-\gamma)$ where $c$ is a payoff form the matrix above, and $\gamma$ determines the curvature. As $\gamma \to 0$, we approach risk neutrality. Focus on rounds with random termination (those are the ones that matter, theoretically). Consider the full cooperation equilibrium under the grim strategy. The equilibrium payoff of a producer at the start of a round is

$$v^* := u(b) + \delta \frac{u(a) + u(b)}{2(1-\delta)}.$$

For simplicity, consider the worst possible case, in which if a producer moves off equilibrium her defection is immediately punished (as if there were public monitoring). This pins down the lowest possible lowerbound discount factor because punishment is immediate, so it gives us a good idea for what happens to the lowerbound threshold under risk aversion. In this case of immediate full punishment, the payoff of the producer who moves off equilibrium is

$$\hat{v}^* := u(y) + \delta \frac{u(x) + u(y)}{2(1-\delta)}.$$

Cooperation is individually optimal if $v^* \geq \hat{v}^*$, which recalling that $a > x$ and $y > b$, can be rearranged as

$$\delta \geq \delta^{**} := 2 \times \frac{u(y) - u(b)}{u(y) - u(b) + u(a) - u(x)}.$$

In the experimental data reported in Section 5 of the paper we used a payoff

matrix where $x = y = 8$ and $a = 20$ and $b = 2$. So we have

$$\delta^{**} := 2 \times \frac{u(8) - u(2)}{u(20) - u(2)}.$$

The termination probability is 0.07 so $\delta = 0.93$. Hence, in the linear case $\gamma = 0$, we would have $\delta^{**} = 2/3$. In the risk-aversion case, we have $\delta = 0.93 \geq \delta^{**}$ for all $\gamma \leq 0.5$ approximately.

Is this lowerbound empirically reasonable? Estimates of CRRA coefficients vary widely depending on subject population, type of experiment (field or laboratory), gender composition and so on (e.g., see Harrison et al., 2009). However, we do note that in experiments with a fixed recruitment fee, such as ours, Harrison et al. (2009) reports an average CRRA coefficient of 0.34, ranging from -0.05 to 0.73 in a 95% confidence interval, while in experiments with low stakes 60% of subjects lay below 0.41 (Holt and Laury, 2002). Hence, full cooperation cannot be ruled out as an equilibrium under some moderate levels of risk aversion.

## B.2 Matlab code to compute discount factors thresholds

To compute the discount factor thresholds $\delta_1$ and $\delta_2$ we used MATLAB R2018a. The algorithm is composed of the following functions: `factorial2.m`, `lmatrix.m`, `tmatrix.m`, `phi_1.m`, `phi_2.m`, `phi_3.m`, `phi_4.m`, `factors.m`.

The description of these functions is as follows: `factorial2.m` returns the double factorial of any input number `n` $\equiv N$; `lmatrix.m` returns the probability matrix of mixed matches (cooperator-defector) for any even number `n` $\equiv N$; `tmatrix.m` returns the transition matrix in Section 3 for any even number `n` $\equiv N$; `phi_`$k$`.m`, with $k$ =1,...,4, returns the value of $\phi_k(\delta)$, given the value of the discount factor (`x` $\equiv \delta$) and the (even) number of players (`n` $\equiv N$) as inputs; `factors.m` returns the lower-bound (`delta_1` $\equiv \delta_1$) and the upper-bound (`delta_2` $\equiv \delta_2$) of the discount factor that solve expressions (6) and (8), respectively, with the latter holding with equality, given `x` $\equiv \alpha$ and `n` $\equiv N$ as inputs. The MATLAB code of each function is reported below.

To run `factors.m`, add all the `*.m` files to the directory `/path/to/MATLAB`, then launch the command `factors(x,n)`, where `x` $\in (0, 1)$ is the parameter $\alpha := \frac{g}{1+l}$ and `n` is the even number $N$ representing the size of the group. All other functions can be launched separately, each returning its own output.

`factorial2.m`

```
function factorial2=factorial2(n)
% returns double factorial of number n

factorial2 = 1;
for i = n:-2:1
factorial2 = factorial2*i;

end
```

lmatrix.m

```
function lmatrix=lambda_matrix(n)
% returns the matrix of probabilities of possible
% mixed matches (cooperator-defector)

L = zeros(n,n);

lmin = 0;
for r = 1:n
lmax = min(n-r,r);
if mod(r,2) == 0
Jeven_upperbound = lmax/2 + 1;
Jeven = zeros(1,Jeven_upperbound);
else
Jodd_upperbound = (lmax+1)/2;
Jodd = zeros(1,Jodd_upperbound);
end

l = lmin:lmax;
Jeven = l(mod(l,2)==0);
Jodd = l(mod(l,2)~=0);

if mod(r,2) == 0
Lambda = zeros(1,Jeven_upperbound);
h = r;
for i=1:Jeven_upperbound
j = Jeven(1,i);
lambda = factorial(j)*nchoosek(r,j)*nchoosek(n-r,j)...
*factorial2(r-j-1)*factorial2(n-r-j-1);
Lambda(i) = lambda/(factorial2(n-1));
L(r,h) = Lambda(i);
h = h+2;
end
else
Lambda = zeros(1,Jodd_upperbound);
h = r+1;
for i=1:Jodd_upperbound
j = Jodd(1,i);
lambda = factorial(j)*nchoosek(r,j)*nchoosek(n-r,j)...
*factorial2(r-j-1)*factorial2(n-r-j-1);
Lambda(i)= lambda/(factorial2(n-1));
L(r,h) = Lambda(i);
h = h+2;
end
end
end

lmatrix=L;

end
```

tmatrix.m

```
function tmatrix=tmatrix(n)
% returns the transition matrix given any even number n

temp = zeros(n,n);
temp=lmatrix(n);
Tmatrix = zeros(n,n);

q=0;
for i = 1:n
jmax = min(n-i,i);
for j=i:min(2*i,n)
    l=j-i;
for h=l: jmax
q=q+(temp(i,h+i))*(nchoosek(h,j-i))*(1/2)^h;
end
Tmatrix(i,j) = q;
q=0;
end
end

tmatrix=Tmatrix;

end
```

phi_$k$.m ($k$=1,2,3,4)

```
function phi_k=phi_k(x,n)
% replace k with a number in {1,2,3,4}
% returns the value of phi_k, given the value of the
% discount factor x and the number of players n


sigmai=zeros(1,n);
phitemp=zeros(1,n);
A=eye(n)-(x)*tmatrix(n);

for j=1:n
sigmai(j)=(n-j)/(n-1);
end

phitemp=(1-x)*inv(A)*sigmai';

phi_k=phitemp(k,1); % with k ∈{1,2,3,4}

end
```

factors.m

```
function factors=factors(x,n)
% returns delta_1 and delta_2, given alpha≡x∈ (0,1) and n even

N=n;
alpha=x;
syms x
sigma_2=(N-2)/(N-1);
Q_Nm2 = tmatrix(N-2);
IO=zeros(1,4);

delta_1=vpasolve(phi_1(x,N)==1-alpha, x, [0,1]);

sol=vpasolve((sigma_2/2)*(x/(1-x))*(Q_Nm2(1,1)...
*(phi_2(x,N)-phi_3(x,N))+Q_Nm2(1,2)*(phi_3(x,N)...
-phi_4(x,N)))==alpha, x, [0,0.9999999]); % =[0,1)
if(isempty(sol))
 delta_2=1;
else
 delta_2=sol;
end

% input-output vector
IO(1,1)= N;
IO(1,2)= alpha;
IO(1,3)= delta_1;
IO(1,4)= delta_2;

fprintf('N = %d\n', N);
fprintf('alpha = %f\n', alpha);
fprintf('delta_1 = %f\n', delta_1);
fprintf('delta_2 = %f\n', delta_2);

%factors=IO;
end
```

# References


Camera, G., C. Deck and D. Porter (forthcoming). Do economic inequalities affect long-run cooperation & prosperity? *Experimental Economics*.

Harrison, G. W., M. I. Laub, and E. E. Rutström. 2009. Risk attitudes, randomization to treatment, and self-selection into experiments. Journal of Economic Behavior & Organization 70(3), 498-507

Holt, C. A, and S. K. Laury. 2002. Risk Aversion and Incentive Effects. *American Economic Review* 92(5), 1644-1655