*Journal of the Economic Science Association* (2026), 1–7
doi:10.1017/esa.2026.10035

economic science association

CAMBRIDGE UNIVERSITY PRESS

EXPERIMENTAL TOOLS

# Distributing fixed resources over many items. An experimental interface

Alessandra Casella[1,2,3] and Olivier Compte[4,5]

[1]Department of Economics and Department of Political Science, Columbia University, New York, NY, USA
[2]National Bureau of Economic Research (NBER), Cambridge, MA, USA
[3]Centre for Economic Policy Research (CEPR), Paris, France
[4]Paris School of Economics, Paris, France
[5]Ecole des Ponts Paris Tech, Marne-la-Vallée, France
**Corresponding author:** Alessandra Casella; Email: ac186@columbia.edu

(Received 5 March 2025; revised 5 March 2026; accepted 26 March 2026)

**Abstract**
We propose an experimental interface designed to elicit the distribution of a budget of resources over a very large number of items. The interface has two related innovations. First, it asks participants to indicate how they allocate the resources by drawing a curve over a stylized representation of the many items. Second, because the curve represents relative, rather than absolute, levels of resources, it remains visually stable when a participant revises any specific allocation, while the program automatically enforces the budget constraint. We describe the application of the interface to a Blotto game with more than 200 items, but the approach can be adapted flexibly to general problems of resource allocations over many items, under a budget constraint.

**Keywords:** Allocation of resources; Blotto games; contests; experimental interface

## 1. Introduction

Economics studies the allocation of scarce resources among competing claims. In many applications, the focus is on the distribution of a budget over multiple items—a basket of goods purchased on a market, in individual choice problems, a multi-unit auction or multiple simultaneous contests, in strategic situations.

In lab experiments, to avoid confusing participants with too complex a task, the number of items under consideration is typically limited to a small number. For example, in their (2015) survey of experiments on different models of contests, Decheneaux et al. cite a total of 201 lab studies. Most study a single contest, but even when the focus is on multiple simultaneous contests, the number of contests is typically three or four, only rarely six or eight.[1] Some influential applications of these models, however, concern problems with large numbers of contested items. A classic example is Myerson (1993): candidates compete to win an election, each has a budget to distribute freely over a continuum of voters, and each voter votes for the candidate who offers the most generous transfer. Myerson's article is a fundamental contribution to voting theory, arguing that politically induced inequality among ex ante identical voters may be the natural result of democratic elections. It is also a very difficult

[1]The larger numbers refer primarily to experiments on Blotto games. See for example, Avrahami and Kareev (2009), Arad and Rubinstein (2012), Chowdhury et al. (2013), Casella et al. (2018).

© The Author(s), 2026. Published by Cambridge University Press on behalf of the Economic Science Association. This is an Open Access article, distributed under the terms of the Creative Commons Attribution-NonCommercial-ShareAlike licence (http://creativecommons.org/licenses/by-nc-sa/4.0), which permits non-commercial re-use, distribution, and reproduction in any medium, provided the same Creative Commons licence is used to distribute the re-used or adapted article and the original article is properly cited. The written permission of Cambridge University Press or the rights holder(s) must be obtained prior to any commercial use.

https://doi.org/10.1017/esa.2026.10035 Published online by Cambridge University Press

theory to test in the field. And yet, it has never been tested in a lab, and neither have equally influential follow-up works that share the focus on politicians competing over voters or electoral districts (Dekel et al., 2008; Lizzeri and Persico, 2001; Lizzeri and Persico, 2005; Sahuguet and Persico, 2006; Snyder, 1989). The difficulty is in designing an experiment that asks participants to allocate resources over a very large number of items, and still keeps the task transparent enough for them to think rationally over the possible choices.

The purpose of this note is to propose an experimental interface that addresses this challenge. The interface is driven by two main ideas. First, in contrast to asking participants for their allocations item by item, as is typically done, participants are asked to draw a curve over a stylized representation of the many items. The curve can be modified freely, with the budget constraint automatically enforced by the computer. Second, the curve depicts *relative* levels of resources over the multiple items, rather than absolute levels. When a participant adjusts the curve, for example by increasing resources on one of the items, the budget is automatically rebalanced so as to satisfy the constraint while maintaining all other relative values constant. As a result, the curve remains visually stable. If the curve showed absolute levels, satisfying the budget constraint would require the whole curve to shift in response to any change, creating a very difficult environment for the subjects.

To our knowledge, the existing experimental interface closest to our goals is the Distribution Builder (Goldstein et al., 2008; Sharpe et al., 2000), an interactive tool used to elicit respondents' preferences over investment portfolios.[2] The Distribution Builder asks subjects to provide a desired distribution of probabilities over future wealth under a cost constraint. The interface computes the least costly portfolio that generates the desired distribution given the prices of available assets, and each subject modifies (by "tatonnement") the desired distribution to meet the cost constraint.

Besides the graphical appearance, our interface and the Distribution Builder have other differences. First, our curve is not a probability distribution; it is a direct allocation of resources. Second, we ask subjects to draw a continuous curve indicating relative levels of expenditure across item values. Thus we call attention to the curve's shape and slope, as opposed to the vector of absolute expenditures, item by item. Third, we implement the budget constraint very differently. In our case, subjects are allowed to draw any curve at all, and the budget constraint is imposed ex post by the computer, when translating relative allocations into absolute levels of expenditure. The Distribution Builder, on the other hand, prevents subjects from finalizing the distribution unless the budget constraint is met. Because asset prices are an important component of the exercise and are not described to the subjects, experimenting with the budget constraints allows subjects to learn them. In our case, there are no hidden prices, and the budget is a straightforward sum of allocated resources.

Because we are not familiar with other experimental interfaces asking participants to express their allocations by drawing a curve and because the range of problems to which the approach could be applied is large, we hope that a detailed description of the interface may be useful to others. The platform was constructed by Destinybit (https://destinybit.com/), a game design company. We describe in the Supplementary Material how to download the program and how to parametrize it for the specific game we studied. However, the logic behind the platform's design is more flexible than the particular application and can be reproduced in different programs. It is such logic, more than the software itself, that is the purpose of this note.

## 2. The experimental interface

The interface was built for the experiment discussed in Casella et al. (2024). For the sake of concreteness, we briefly describe the game here. We study a Blotto game where two opponents are each endowed with a budget of resources $R_i$, to be distributed over a continuum of battlefields. The battlefields have different values $v \in \mathcal{V}$, drawn independently from a common distribution

[2]We thank an anonymous referee for bringing the Distribution Builder to our attention.

https://doi.org/10.1017/esa.2026.10035 Published online by Cambridge University Press

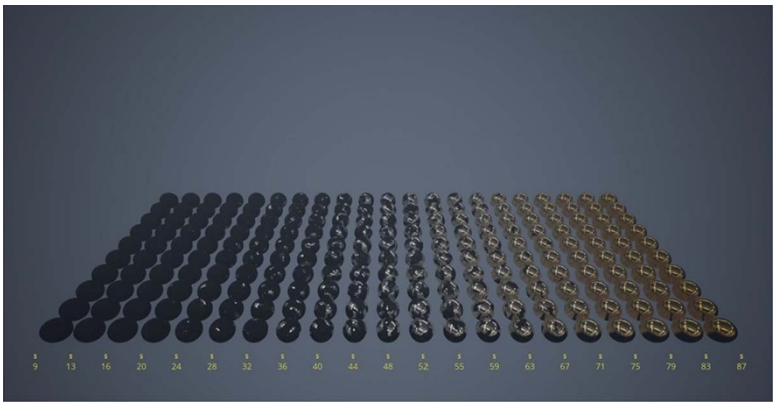



**Fig. 1** Experimental interface: opening screen. $f(v)$ uniform; 210 marbles

$f(v)$. Independence holds both across battlefields, for given player, and across the two opponents, for given battlefield. Each battlefield is won by the player who allocates to it more resources. A player's strategy is a resource allocation function $b_i(v)$ that maps the battlefields' values into resources allocated to each value, under the constraint that the resource constraint be satisfied, or $\int_{\mathcal{V}} b_i(v)f(v)dv = R_i$. The object of interest is the allocation function $b_i(v)$. The theory delivers specific predictions on the shape of such function in equilibrium; the challenge was eliciting such a function experimentally.[3]

In the experiment, we framed the battlefields as assets to be defended and represented them as marbles of different values. All marbles a player owns eventually clash with another player's marbles in one-to-one duels, but before duels occur, players make their marbles "stronger" by endowing them with defenses from a known budget of defenses. In each duel, the marble with lower defenses is destroyed, the stronger one survives, and its owner earns the marble's value as points. As mentioned, the Appendix provides a link for downloading the program, as well as instructions on how to make some modifications to the original game, for example by changing the distribution $f(v)$, or allowing for asymmetric budgets $R(i)$, or introducing correlation in values.

At the start of the experiment, each participant sees the marbles with which each is endowed. These are the assets, of different values, the participant will have to defend. With $f(v)$ Uniform, the first screen corresponds to Figure 1.

To approximate the continuous distribution, players must be endowed with as many marbles as possible. But when the number is large, handling each marble separately (appreciating its value and choosing the desired level of defenses) is infeasible. Marbles must then be organized in classes. In practice, players are faced with 21 classes of marbles, organized in columns, with each column corresponding to a different value, and all marbles in the same column having the same value. Thus the marbles can take any of 21 different values, roughly suggested by their color, from black to gold, and reported more precisely by the number at the bottom of each column. In the case of a Uniform distribution, as in the example of Figure 1, all columns have the same number of marbles (here, 10), and thus there is a total of 210 marbles. All marbles will fight duels.[4]

[3]In Blotto games, the total amount of resources to be distributed is fixed. More generally, in problems with a budget constraint, the resources' residual use would be depicted on the platform as an additional item.

[4]Because each marble is matched in duel against a random opponent, the multiplicity of marbles of the same value works to ensure that the points earned by the player from a class of value $v$ approximate ex ante expected earnings from a random match for a marble of value $v$.

https://doi.org/10.1017/esa.2026.10035 Published online by Cambridge University Press

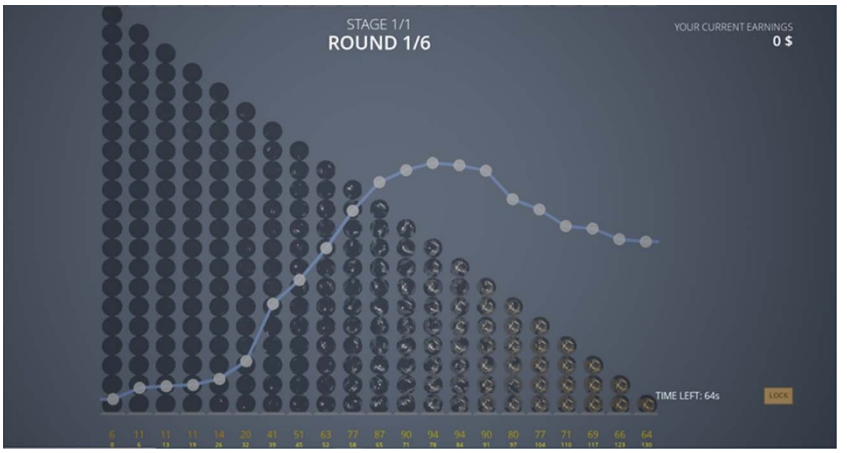



**Fig. 2** Experimental interface: distribution of defenses. Example with $f(v)$ triangular; 231 marbles

A player's task is the allocation of defenses to the player's marbles. Defenses are not represented through objects but graphically, through a curve each player draws, the literal experimental counterpart of the $b(v)$ bidding function of the theoretical model. We asked each participant to allocate defenses by moving the cursor of the computer mouse smoothly over the image of the marbles. All marbles in the same column receive the same amount of defenses, represented by the height of the curve over that column. Figure 2 is an example we used in the instructions for the case of a triangular distribution $f(v)$.

Although only 21 different values are thus relevant when allocating defenses, the game is much more intuitive if presented as if the value distribution were continuous. When asked to draw a curve, players focus on the shape of the curve, that is, on the slope of the allocation function—the change in defenses as the values of the marbles increase—as opposed to choosing the exact numerical amount of defenses assigned to each of the 21 values. This short video gives an idea of the interface. We describe the different steps in the next section.

## 3. Eliciting the defense allocation function

Asking participants to draw a curve creates two related challenges. First, the visualization of different allocations needs to be somewhat robust to the scale of the graph: the image on the screen must remain legible whether the player devotes all defenses to a single column, or divides them less asymmetrically. We set $R_i = 10,000$ to allow for a fine enough variation in assigned defenses while still expressing them in integer numbers. As a result, however, the absolute value of defenses assigned to each marble could vary widely, creating difficulties with the visualization. In the Uniform case, for example, with $R_i = 10,000$, the amount of defenses assigned to a marble could vary between 0 and 1,000 (taking into account that all marbles in the same column must receive the same defenses). But then the difference between, say, 25 and 50, would be impossible to see, even though one number is twice the other. Second, the defense budget must be satisfied. If we asked participants to draw absolute resource allocations, then revising the defenses assigned to one column would force other allocations to change as well, to enforce the budget constraint. The global change in the curve would be sure to generate confusion. Asking participants to report graphically the *relative* allocation of defenses per column addresses both problems.[5]

[5]The difficulty of seeing small differences in allocations is reduced by an order of magnitude. It remains when allocations are very small and quite similar, but also economically not very meaningful.

https://doi.org/10.1017/esa.2026.10035 Published online by Cambridge University Press

Call $Rmax_i$ the maximal defense assigned by player $i$ to any marble (note: any marble, not any column, since, depending on the distribution $f(v)$, columns can have different numbers of marbles, as they do in Figure 2). $Rmax_i$ provides the anchor against which relative allocations are calculated. Its value, in absolute terms, is calculated by the computer for any allocation of reserves, so as to satisfy the budget constraint, and adjusts with any change to the allocation.

The vertical axis corresponds to fractions of $Rmax_i$ and ranges between 0 and $100\%Rmax_i$. In Figure 2, the curve is three times higher in column 7 relative to columns 2, 3 and 4, and thus the curve assigns three times as many defenses to any marble in column 7, relative to any marble in columns 2, 3, and 4. The larger number at the foot of each column indicates the absolute defenses assigned to each marble in the column, while the smaller number reminds the player of the value of the marbles in the column.

At the start, before actively allocating defenses, the subjects see a horizontal line over all columns, an initial allocation curve assigning equal defenses to all marbles.[6] During play, participants modify the curve. Participants modify a point on the curve by clicking on it and raising or lowering it.

The curve is anchored by points we call *anchors*. A point on the curve becomes an anchor by being clicked on, giving the visual impression of "grabbing" the curve at that point. Only points corresponding to a column can be anchors, but because the columns fill compactly the whole screen, the player's perception is that any point on the curve can be clicked on and modified. In addition, although each individual point can only be moved vertically, the density of columns on the screen means that when drawing a continuous curve the experience approximates closely the sense of drawing without constraints.[7] In practice, as a participant draws a curve by moving the cursor of the mouse freely over the marbles, the points corresponding to all columns typically become anchors.

Anchors matter because when participants raise or lower one point, the curve only changes between the two closest anchors, on the point's either side: the segments of the curve between the point and its closest anchors on the point's two sides adjust by linear interpolation. The other segments, outside the two closest anchors, do not move. When, as most likely, points corresponding to all columns have become anchors, modifying the curve over one column has no effect on the height of the curve over any other column. To reiterate our main point, this is possible because the curve represents relative defenses. If it represented absolute defenses, the entire curve would move in response to any revision because all allocations of absolute defenses (including $Rmax_i$) would have to change to satisfy the budget constraint.

To calculate the amount of absolute defenses assigned to any column, the program solves two problems. First, the program determines the defenses assigned to each marble, expressed in percentage of $Rmax_i$, by linear interpolation between any two anchors. Second, the program calculates $Rmax_i$.

The first step corresponds to the visual changes of the curve, as the player makes any adjustments. For example, if the vertical height of the allocation curve is set at 20 at position (value) 75, and at 100 at position 100, if both are anchors, and there is no other anchor between them, then at position 80 the allocation curve is set automatically at $20 + [(100 - 20)/(100 - 75)] \times 5 = 36$. If we call $x$ the horizontal position (recall that $x$ is the value of marbles at the corresponding column) and $y$ the height, then:

$$y_m = y_l + \left(\frac{y_r - y_l}{x_r - x_l}\right)(x_m - x_l)$$

where $m$ denotes the point of interest, intermediate relative to the anchors to the left ($l$) and to the right ($r$). Here $x_l = 75$, $x_r = 100$, $y_l = 20$, $y_r = 100$, and $x_m = 80$. Hence $y_m = 36$.

[6] In this case, and in this case only, the height of the curve is irrelevant, and participants are warned of this and explained why. All marbles have defenses equal to $100\%Rmax$.

[7] Under the limitation that the curve can only appear on each column once: each column can have only one allocation of defenses.

https://doi.org/10.1017/esa.2026.10035 Published online by Cambridge University Press

If the subject decides to move the curve at position 80, then the height at position 80 is determined by the subject, while the heights strictly between position 75 and position 100 (the two closest anchors on the two sides) are determined by the equation above. If $y_{80} = 50$, for example, then:

$$y = 20 + \left(\frac{50-20}{80-75}\right)(x-75) \text{for all points between positions 75 and 80}$$
$$y = 50 + \left(\frac{100-50}{100-80}\right)(x-80) \text{for all points between positions 80 and 100.}$$

The curve does not change at or below position 75. This is the program's first step.

As a second step, given the defenses assigned to each marble as proportion of $Rmax_i$, the program calculates $Rmax_i$ using the constraint that all defenses must equal the total budget. Given the value of $Rmax_i$, the program then translates all relative defenses into absolute values, reported numerically at the bottom of each column.

If we call $n_c$ the number of marbles in column $c$, $C$ the total number of columns (21 in our implementations), and $y_c$ the height of the curve at column $c$, then $Rmax_i$ must solve:

$$\sum_{c=1}^{C} n_c \frac{y_c}{100} Rmax_i = R_i$$

where $R_i$ is $i$'s total available defense budget.

In the example of Figure 2, the total number of marbles is 231, divided in 21 columns with $\{21, 20, ..., 1\}$ marbles each. At the opening screen, $y_c = 100$ for all columns; if $R_i = 10,000$, we have $231(100/100)Rmax_i = 10,000$, or, rounding up, $Rmax_i = 43$. Since at the opening screen $y_c = 100$ (or 100% $Rmax_i$), every marble is allocated absolute defenses equal to 43. Suppose instead, for example, $y_c = 20$ for the first 10 columns, and 100 for columns 11-21. Then: $\left(\sum_{z=12}^{21} z(20/100) + \sum_{z=1}^{11} z(100/100)\right) Rmax_i = 10,000$, or, after rounding, $Rmax_i = 101$. Each marble in the first 10 columns is allocated defenses just above 20, and each marble in columns 11-21 defenses equal to 101.

As the player manipulates the curve, all larger numbers at the foot of each column, representing the absolute amount of defenses assigned to each marble in the column, necessarily change (to respect the budget constraint), while the curve remains stable beyond the two anchors closest to the point the player has modified. The program stores, for each player, the vector of 21 $x, y$ values along the final allocation curve, as well as the history of revisions made by the player.

The computer interface was designed to maximize the sense of drawing a continuous curve and little practice is sufficient to generate the allocation curve by moving the mouse smoothly over the profile of all marbles. The curve can be manipulated easily—for example, players can delete a point by right-clicking on it (and the height of the allocation curve at that point is then interpolated linearly from the two nearest anchors)[8]—and at any modification the constraint that the entire defense budget be allocated is imposed automatically by the computer. The design allows players to draw a defense allocation function that is intuitive, uses the full budget, and remains stable as details are fine-tuned.

**Supplementary material.** The supplementary material for this article can be found at https://doi.org/10.1017/esa.2026.10035.

**Funding.** We thank Gian Paolo Vernocchi and Simone Daminato at Destinybit for their programming expertise and creativity, Jean Francois Laslier for inspiring conversations on the initial design, and the Institute for Social and Economic Research and Policy (ISERP) at Columbia University for financial support.

[8]See the example at time 0:28 in the video, where the curve is dragged downward in the sixth column from the right, but snaps back upward when that point has been right-clicked and has stopped functioning as an anchor.

https://doi.org/10.1017/esa.2026.10035 Published online by Cambridge University Press

## References

Arad, A., & Rubinstein, A. (2012). Multi-dimensional iterative reasoning in action: The case of the Colonel Blotto game. *Journal of Economic Behavior and Organization*, *84*(2), 571–585.

Avrahami, J., & Kareev, Y. (2009). Do the weak stand a chance? Distribution of resources in a competitive environment. *Cognitive Science*, *33*(5), 940–950.

Casella, A., Laslier, J. F., & Mace', A. (2018). Democracy for polarized committees: The tale of Blotto's lieutenants. *Games and Economic Behavior*, *106*, 239–259.

Casella, A., Compte, O., & Si, S. (2024). Dealing with complex games: Methodological notes and an experiment. Presented at the 6th Columbia Conference in Economic Theory, September 12–13.

Chowdhury, S., Kovenock, D., & Sheremeta, R. (2013). An experimental investigation of Colonel Blotto games. *Economic Theory*, *52*(3), 833–861.

Dechenaux, E., Kovenock, D., & Sheremeta, R. (2015). A survey of experimental research on contests, all-pay auctions and tournaments. *Experimental Economics*, *18*(4), 609–669.

Dekel, E., Jackson, M., & Wolinsky, A. (2008). Vote buying: General elections. *Journal of Political Economy*, *116*(2), 351–380.

Goldstein, D., Johnson, E., & Sharpe, W. (2008). Choosing outcomes versus choosing products: Consumer-focused retirement investment advice. *Journal of Consumer Research*, *35*(3), 440–456.

Lizzeri, A., & Persico, N. (2001). The provision of public goods under alternative electoral incentives. *American Economic Review*, *91*(1), 225–239.

Lizzeri, A., & Persico, N. (2005). A drawback of electoral competition. *Journal of the European Economic Association*, *3*(6), 1318–1348.

Myerson, R. (1993). Incentives to cultivate favored minorities under alternative electoral systems. *American Political Science Review*, *87*(4), 856–869.

Sahuguet, N., & Persico, N. (2006). Campaign spending regulation in a model of redistributive politics. *Economic Theory*, *28*, 95–124.

Sharpe, W., Goldstein, D., & Blythe, P. (2000). The distribution builder: A tool for inferring investor preferences. Available at https://web.stanford.edu/wfsharpe/art/qpaper/qpaper.html. Stanford CA.

Snyder, J. (1989). Election goals and the allocation of campaign resources. *Econometrica*, *57*(3), 637–660.

**Cite this article:** Casella, A., & Compte, O. (2026). Distributing fixed resources over many items. An experimental interface. *Journal of the Economic Science Association*, 1–7. https://doi.org/10.1017/esa.2026.10035

https://doi.org/10.1017/esa.2026.10035 Published online by Cambridge University Press