Pseudocode and Algorithms for Computer Simulations of Democratically Planned Economies

Mitchell Szczepanczyk

Abstract

The claim that "there is no third way" besides the economic models of capitalism and communism has faced a challenge from a new and growing body of research into a "third way" economic paradigm known as democratic planning. In this paper, we explore one of these democratic planning models – Robin Hahnel and Michael Albert's model of a participatory economy, focusing in particular on its allocation mechanism – a non-market, non-command-planning procedure known as "participatory planning." This procedure has recently been implemented in computer programs to explore its feasibility, the encouraging results of which have been published elsewhere and which we summarize here. But I present here for the first time the detailed algorithms and related pseudocode powering all of these computer programs for others to consider, examine, and build as their own programs. I also describe future directions for this avenue of research.

Keywords
Participatory Economy, Democratic Planning, Economic Simulation Algorithms

Introduction

Most of the twentieth century was dominated by two main economic systems: capitalism (characterized by markets and limited liability corporations), and Communism (characterized by command planning, also known as central planning with a central-planning authority). Despite the various and abundant problems in these two main economic systems, the guiding wisdom was that all complex industrial economies had to choose between one of these two systems. The economic historian Alec Nove wrote in no uncertain terms that "there is no third way".[1] For many, to broach the topic of exploring other economies outside of these terrible two, even in theory, seems fanciful to the point of absurdity.

And yet, beginning in 1990 there has been research into a hypothesized "third way" economy that is neither Communist nor capitalist, which goes by the generic name of "democratic planning". In democratic planning, workers and consumers organize their economic affairs democratically, but without markets, without corporations, and without a command planning authority. Indeed, a handful of different models in this democratic planning vein have been proposed.[2] In light of these developments, those who followed Nove's dictum that "there is no

---

[1] Nove, 44.

[2] Five models of democratic planning are outlined and discussed in the appendix of Hahnel (2021). In this book Hahnel also discusses the problems with both market economies and command-planning economies in chapters two, three, and four. Disclosure: The author of this paper is one of the contributors to this book.

third way" have apparently dropped that objection and have instead raised new objections. Among them: Is such an economy feasible? Is it a practical possibility? That is, would it actually work in the real world?

Advocates of democratic planning have sought to address such objections, and in recent years have taken to using computers and computer programs to do so. In this paper, we explore in depth one of these democratic-planning models, referred to as a "participatory economy". We focus on the algorithms and pseudocode that have been devised in computerized simulations to explore its main organizational function -- its non-market, non-command-planning allocation mechanism, which goes by the name "participatory planning" – in order to address the objection that a democratically-planned economy is not a practical possibility. With a roadmap of the algorithms and pseudocode, we encourage others to build and explore their own implementations of this model in any computer programming language.

A Participatory Economy in outline

There are four main institutional components of the "third way" democratically-planned economy known as a participatory economy:[3]

1. *Division of labor*. Both capitalism and communism have hierarchical divisions of labor, where few perform the more empowering and desirable work while the vast majority perform less empowering and less desirable work. In a participatory economy, work is balanced so that all participants have a comparable mix of empowering and desirable work.

2. *Remuneration*. In twentieth century "communist" economies people were often paid according to how much their human capital contributed to the value of output. People in capitalist economies are paid according to market power and how much their human *and* physical capital contribute to the value of output. However, people in a participatory economy are paid according to the effort and sacrifice expended in socially valued labor, tempered by need.

3. *Institutional sovereign*. In a communist economy the planning authority is "sovereign" (i.e., it is recognized as the ultimate authority in the economy). In a capitalist economy limited-liability corporations are "sovereign." In a participatory economy worker councils and consumer councils in which all members have an equal vote, as well as federations of worker and consumer councils, are "sovereign."

4. *Allocation mechanism*. A capitalist economy allocates goods and services by means of markets, while a communist economy does so by means of command planning. A participatory economy, as already mentioned briefly, uses an allocation mechanism known as "participatory

---

[3] In addition to the aforementioned Hahnel (2021), other books that describe and elaborate on the participatory economy model include Hahnel (2022), Albert (2021), Hahnel and Wright (2016), and Albert and Hahnel (1991). The model is also described online at many websites, including https://www.participatoryeconomy.org

planning"[4] where worker councils propose what they want to produce and how they want to produce it, consumer councils propose what they want to consume, and councils then revise those proposals in a series of iterations until excess demands for different goods, categories of labor, and inputs from the natural environment are eliminated, i.e. until a "feasible plan" is reached. More specifically:

The actors in participatory planning are:
A. Worker councils, and federations of worker councils.
B. Neighborhood consumer councils, and federations of consumer councils.
C. The best indicators available of social and opportunity costs of goods and services, referred to as "indicative prices".
D. A procedure to announce and adjust indicative prices, referred to in the literature as an "iteration facilitation board" (IFB).

The annual participatory planning procedure is as follows:
1. The IFB announces current estimates of indicative prices for all goods and services in the economy.
2. Consumer councils make proposals for what they propose to consume, while worker councils make proposals for what they propose to produce, and what inputs they would require to do so.
3. If all excess demand from consumer councils' proposals are met by production proposals from worker councils, we have an allocation plan which economists call "feasible," i.e. it could be carried out. If not, the IFB adjusts indicative prices up or down based on differences in supply and demand. Councils then revise their proposals based on the new information from the IFB and the process is repeated until all excess demands are eliminated.

It is this annual participatory planning procedure which has been the focus of recent research using computational, agent-based models, and which is also the focus of the present paper. One key issue this research addresses is what has been called the "practicality question" -- how many iterations would a participatory economy require in order to arrive at a feasible economic plan? If the procedure requires hundreds of iterations for a single annual economic plan, that's certainly not very practical. Proponents of participatory planning have argued that there is no reason to assume that the number of iterations required would be excessive, and therefore impractical. But until recently there was no empirical evidence to support this conclusion.

However, we now have *some* empirical evidence, thanks to the development of computerized models implementing the annual participatory planning procedure. So far the evidence seems to confirm what participatory economy advocates conjectured – that the number of iterations necessary for a participatory planning procedure to converge to a feasible plan is quite practical. Over many experiments, my fellow researchers and I discovered that on average it took only six

[4] More precisely, what is described here, and what we simulated in a computer program, is the participatory *annual* planning procedure, which determines what is produced and consumed during a year. Participatory procedures for generating investment and long-run development plans of different kinds have also now been proposed. See Hahnel (2021) and Hahnel (2022).

iterations, with very little variation.[5] We now turn to the details of these computerized models as algorithms and their pseudocode.

Economic and Mathematical Preliminaries

The actors in a participatory planning procedure include worker councils and consumer councils. In a mathematical model, we can represent the behavior of worker councils using production functions and we can represent the behavior of consumer councils using utility functions. In particular, we use Cobb Douglas functions to represent these councils. The general form for a production function is:

$$\mathrm{Y} = \mathrm{AL}^{\mathrm{b}}\mathrm{K}^{\mathrm{a}}$$

where

Y is output
L is labor input
K is capital input
a and *b* are output elasticities (the relationship of percentage change between input and output)
A is total factor productivity (a representation of what goes into the output that isn't labor nor capital)

We adapt this prototypical Cobb Douglas function for our simulated participatory planning procedure as follows:

$$z = qx^a r^b l^c e^d$$

where:

z is output
x is a vector of intermediate inputs (goods used in the production of other goods)
r is a vector of natural resource inputs
l is a vector of different kinds of labor inputs
e represents the effort level expended by workers
a, *b*, *c*, and *d* give the elasticities for each input
q is the total factor productivity

We define the well-being generated by a worker council as follows:

$$WB(x, r, l, e) = \left[p_z q x^a r^b l^c e^d - [p_x x + p_r r + p_l l]\right] - e^f$$

[5] A full presentation of the results of large-scale experiments investigating various aspects of the participatory planning procedure can be found in chapter 9 of Hahnel (2021).

where p_Z represents the social value of the worker council's output, and p_X, p_r and p_l represent the opportunity costs of using its various inputs, the "indicative prices" previously explained.

Therefore, the first term represents an estimate of the value of worker council output to others, the second term represents an estimate of the opportunity cost to society of all the inputs used by the worker council, and the final term represents the negative utility to the workers in the council from the effort they expend. Since income for members of a worker council depend positively on the quantity in large brackets and negatively on *e*, the members will presumably want to choose *x,r,l,* and *e*. to maximize the difference between the term in large brackets and the final term.

We then convert these formalisms into equations suitable to code. For this purpose, we use Wolfram Mathematica[6] to generate the equations. The following is an example in Wolfram Language for a production function with two intermediate goods. We define a set of variables we wish to calculate. Following the convention above, we first declare a list of variables for output, intermediate goods, and effort:

```
vars = {z, x1, x2, ef};
```

We then define a function y which takes `vars` as input and computes a Cobb Douglas production function involving the total factor of productivity and elasticities:

```
y[vars] := a * x1^b1 * x2^b2 * ef^c;
```

We have a companion function `dy` to take the derivative of y, using the Wolfram function `D` for derivative in relation to a given intermediate good and accompanying elasticity. (We take the derivative since we want to maximize the overall net well-being of the worker council.)

```
dy[xj] := D[y[vars], xj];
```

We then use the Mathematica `Solve` function to solve a system of equations for the variables in `vars` as follows:

```
Solve[
   z == a * x1^b1 * x2^b2 * ef^c &&
   p == λ &&
   p1 == λ * dy[1] &&
   p2 == λ * dy[2] &&
   k = S * ef^k-1 == λ * D[y[vars], ef],
   vars
];
```

where

---

[6] https://www.wolfram.com

`p1` and `p2` represents the price of each input
`p` and λ refer to the price of each output
`k` refers to the disutility of effort (the dissatisfaction associated with work), calculated with an accompanying coefficient S

The solution for this set of equations involving just two inputs appears below (written in the Clojure programming language) in Figure 1. The result is quite complicated and grows only more complicated with more inputs. For our research, we limited the number of inputs in our production function to a maximum of eight. However, I was able to generalize the production functions to any number of inputs, thanks to the regularity of the patterns in the solutions, so it is possible to extend the model accordingly.

```
(defn solution-two-inputs [a s c k ps b λ p-i]
  (let [[b1 b2] b
        [p1 p2] (flatten ps)
        output (Math/pow Math/E (/ (+ (- (* k (Math/log a))) (- (* b1 k (Math/log b1)))
(- (* b2 k (Math/log b2))) (- (* c (Math/log c))) (* c (Math/log k)) (* b1 k (Math/log p1))
(* b2 k (Math/log p2)) (* c (Math/log s)) (- (* c (Math/log λ))) (- (* b1 k (Math/log λ)))
(- (* b2 k (Math/log λ)))) (+ c (- k) (* k b1) (* k b2))))
        x1 (Math/pow Math/E (/ (+ (- (* k (Math/log a))) (* c (Math/log b1)) (- (* k (Math/log b1)))
(* b2 k (Math/log b1)) (- (* b2 k (Math/log b2))) (- (* c (Math/log c))) (* c (Math/log k))
(- (* c (Math/log p1))) (* k (Math/log p1)) (- (* b2 k (Math/log p1))) (* b2 k (Math/log p2))
(* c (Math/log s)) (- (* k (Math/log λ)))) (+ c (- k) (* k b1) (* k b2))))
        x2 (Math/pow Math/E (/ (+ (- (* k (Math/log a))) (- (* b1 k (Math/log b1)))
(* c (Math/log b2)) (- (* k (Math/log b2))) (* b1 k (Math/log b2)) (- (* c (Math/log c)))
(* c (Math/log k)) (* b1 k (Math/log p1)) (- (* c (Math/log p2))) (* k (Math/log p2))
(- (* b1 k (Math/log p2))) (* c (Math/log s)) (- (* k (Math/log λ)))) (+ c (- k) (* k b1) (* k b2))))
        effort (Math/pow Math/E (/ (+ (- (* (Math/log a))) (- (* b1 (Math/log b1)))
(- (* b2 (Math/log b2))) (- (* (Math/log c))) (* b1 (Math/log c)) (* b2 (Math/log c))
(* (Math/log k)) (- (* b1 (Math/log k))) (- (* b2 (Math/log k))) (* b1 (Math/log p1))
(* b2 (Math/log p2)) (* (Math/log s)) (- (* b1 (Math/log s))) (- (* b2 (Math/log s)))
(- (* (Math/log λ)))) (+ c (- k) (* k b1) (* k b2))))
        [input-qs nature-qs labor-qs] (assign-new-proposal p-i [x1 x2])]
    {:output output
     :x1 x1
     :x2 x2
     :effort effort
     :input-quantities input-qs
     :nature-quantities nature-qs
     :labor-quantities labor-qs}))
```

Figure 1- The resulting output, effort, and corresponding goods in our model for a workers' council involving two inputs.

The utility function to simulate a consumer council in the participatory planning procedure is more brief and nowhere near as complicated:

$$WB(y1, y2) = y_1^a y_2^b$$

subject to

$$I_{cc} = \left(p_{y1} y_1^a\right) + \left(p_{y2} y_2^b\right)$$

where

y1 is a vector of the private goods consumed by a consumer council
y2 is a vector of the public goods consumed by a consumer council
a and *b* are the corresponding vectors of elasticities
p_{y1} and p_{y2} are the corresponding vectors of indicative prices for private and public goods
I_{cc} represents the income earned by the members of the consumer council

In short, the well-being of a consumer council is represented by the utility they enjoy from consuming whatever bundle of private and public goods they consume. The constraint reminds us that a consumer council cannot spend more than it has in its budget.

Algorithm and Pseudocode

We can now articulate an algorithm to simulate a participatory planning procedure. We endeavor to track five broad categories of goods -- private goods, intermediate goods, public goods, natural resources, and categories of labor. The algorithm in broad strokes is as follows:

1. Generate all worker councils, consumer councils, and indicative prices, and load them all into memory.
2. For each worker council: Solve its well-being optimization problem -- update its output, effort, and demands for inputs based on the latest indicative prices.
3. For each consumer council: Solve its well-being optimization problem -- update its demands for public and private goods based on the latest indicative prices.
4. Calculate the new excess demands for all goods, natural resources, and categories of labor, and apply a price adjustment formula (about which more below) to update all indicative prices.
5. Increase our iteration counter by one.
6. Check all excess demands. Do they all fall within a prespecified threshold? If so, stop. If not, return to step 2 and repeat.

We arrived at a price adjustment rule with some trial-and-error which seemed to be effective in helping the economy converge to a plan. The price adjustment formula we used is as follows:

$$\mathrm{w} = \mathrm{v}(1.05 - 0.5^{\mathrm{v}}), \text{ except when } \mathrm{v} > 0.25 \text{ then } \mathrm{v} = 0.25$$

where: *w* is the percentage change in the price of a good for the next iteration, and
v is the percentage excess supply for a good in the iteration just completed

We implemented the following algorithm in the multi-agent programmable environment Netlogo[7], as well as in Clojure (a list-processing functional language which compiles to Java bytecode for the Java Virtual Machine)[8] and in Clojurescript (a variant of Clojure which

[7] https://ccl.northwestern.edu/netlogo/

[8] https://www.clojure.org

compiles to Javascript for the World Wide Web)[9]. In all these instances, we implemented the detailed procedure below outlined in pseudocode.

1. Instantiate worker councils. Each worker council is a key-value data structure incorporating the following values, based largely on the well-being function for worker councils.

* industry (a simple identifier - 0 for private goods, 1 for intermediate goods, 2 for public goods)
* product (another simple identifier, 0 or 1)
* s (coefficient for disutility of effort)
* du (exponent for the disutilty of effort)
* c (effort elasticity)
* effort (defined with a starting value of 0.5)
* a (the total factor of productivity - a floating-point value within a given range, say 4 and 6)
* labor-quantities (starting with a very simple vector of vectors: [[0]])
* production-inputs (a set of vectors to represent intermediate goods, natural resources, and kinds of available labor -- a typical example could be [[6 7] [8] [1 2]])
* input-exponents (the elasticities for intermediate goods: for a set of two intermediate goods a simple example could be [0.16, 0.15])
* nature-exponents (the elasticities for natural resources: for a single entry, an example could be [0.12])
* labor-exponents (the elasticities for available labor: for a set of two, an example could be [0.10 0.11])
* output (defined to start at a value of 0.5)

There is one important constraint to every worker council which is worth noting here: The combined set of exponents (intermediate input-exponents, nature-exponents, and labor-exponents) when added together must be less than one for any worker council. Violating this rule would lead to increasing returns to scale. We intentionally violate this rule later to test our procedure for robustness, as explained below.

2. Instantiate consumer councils. Each consumer council is also a key-value data structure incorporating the following values based largely on the well-being function for consumer councils.

* number-of-consumers (defined with a starting value of 10)
* private-good-demands (the private goods being consumed, comprising a list like [0 0 0 0 0])
* public-good-demands (the public goods being consumed, also a list like [0 0 0 0 0])
* private-good-exponents (the elasticities for the private goods, a list of values like [0.03 0.02 0.06 0.04 0.05])
* public-good-exponents (the elasticities for the private goods, also a list of values like [0.02])
* cy (the total factor of utility, defined with a value of 1)

3. In the memory state of the program, we instantiate the following values.

[9] https://clojurescript.org

* The list of available private goods (for a list of ten goods, for example, a list of numbers 1 through 10)
* The list of available public goods (also a list 1 through N for a value N)
* The list of available intermediate goods (also a list 1 through N for a value N)
* The list of available natural resources (also a list 1 through N for a value N)
* The list of available categories of labor (also a list 1 through N for a value N)
* The list of prices for each of the available private goods, public goods, intermediate goods, natural resources, and labor categories (in one instantiation, we set each good to a price of 700)
* The surpluses for each of the available private goods, public goods, intermediate goods, natural resources, and labor categories.
* The supplies of natural resources (assigned in one instantiation at a value of 1000 to start).
* The supplies of categories of labor (assigned in one instantiation at a value of 1000 to start).
* An iteration counter, set at zero to start.

There are various other statistics that can be initialized. These include price deltas (the percentage change in price across iterations), the gross domestic product of the economy, the Gini index (a measure of income inequality), and so on. Various past instances of code have computed and displayed subsets of these statistics.

4. For each worker council, solve its optimization problem. The function which does this, named `produce`, takes as input the indicative prices and the following components:

a (the total factor of productivity)
s (the coefficient for disutility of effort)
c (effort elasticity)
k (synonym for the exponent of the utility of effort)
ps (the set of indicative prices)
p-i (the production inputs)
b (the exponents for the production inputs)
λ (the price for a given worker council's product and industry).

The output of the function is the worker's council revised output, revised effort, and revised demands for inputs.

5. For each consumer council, solve its optimization problem. The function which does this, named `consume`, takes as input the indicative prices for public goods, and the indicative prices for private goods, and the following components: income, private-good-demands, private-good-exponents, public-good-demands, public-good-exponents, as well as the total number of consumer councils in the economy.

For a given consumer council, we proceed as detailed below.

For each private good, the new demand for that private good is computed as follows:

(income * private-good-exponent) /
[(sum of all private-good-exponents and public-good-exponents) * (private-good-price)]

For each public good, the new demand for that public good is computed as follows:

(income * public-good-exponent) /
[(sum of all private-good-exponents and public-good-exponents) * (public-good-price / number-of-consumer-councils)]

We divide the price by the number of consumer councils whose members are also consuming the public good, while we assume that every consumer of a public good must pay only his or her proportionate share of its cost.

6. Update the surpluses (the difference between supply and demand) for all goods, natural resources, and categories of labor.

7. Update all the indicative prices in the economy as follows:

For each good within each of the five categories outlined:
* Compute its surplus (supply minus demand)
* Compute its new price adjustment, which we defined as [v * (1.05 - 0.5^v)]. We adapt this in pseudocode to be:

1.05 - 0.5^[|(surplus * 2)| / (supply + demand)]

* Multiply this price adjustment (termed a price delta) to the corresponding price adjustment of the previous round and take its absolute value. Then take the smaller of this product and the price adjustment just computed. Return this result or 0.001 whichever is larger.

* If the surplus is positive (if supply is greater than demand), take that result and subtract it from one (so that [1 - new-price-adjustment]) then multiply by the price of the corresponding good.

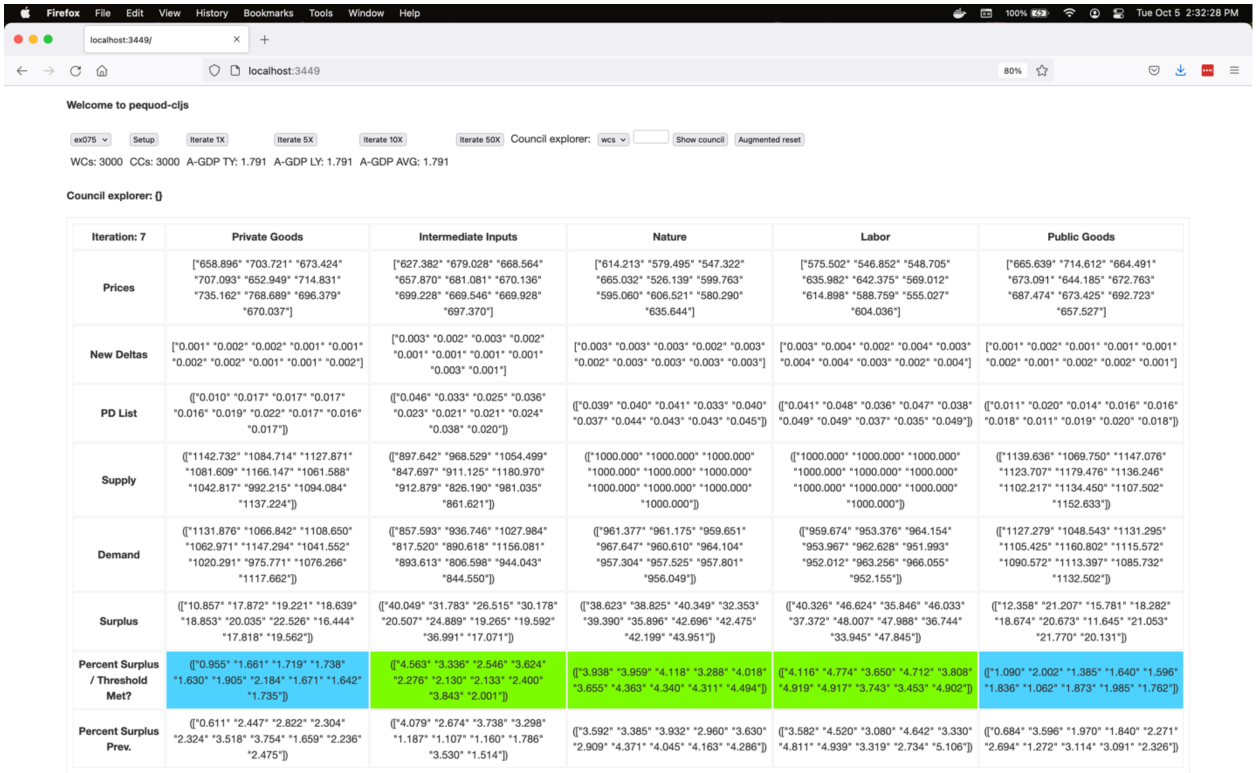
* If the surplus is negative (if supply is less than the demand), take that result and add it to one (so that [1 + new-price-adjustment]) then multiply by the price of the corresponding good.

* If the surplus is zero (supply and demand are equal), return the previous price.

8. We increase our iteration counter by one and then check to see if each surplus falls within a prescribed threshold. The algorithm calls for the planning procedure to halt if the threshold is met. However, for the purposes of exploration, we didn't always implement a halting mechanism, and one strictly speaking isn't necessary in the code. If the threshold is not met, we return to step four above and repeat.

--

In one implementation where we display various components of the procedure, we color-coded the resulting surpluses as a convenience. See Figure 2 for a screenshot of such an implementation. As the surpluses for a given category of goods approached zero, the color of the display showing the surpluses changed from red (greater than 20% surplus), to orange (10%-20% surplus), to yellow (5%-10%), to green (3%-5%) to blue (3% surplus or less).



Iteration: 7	Private Goods	Intermediate Inputs	Nature	Labor	Public Goods
Prices	["658.896" "703.721" "673.424" "707.093" "652.949" "714.831" "735.162" "768.689" "696.379" "670.037"]	["627.382" "679.028" "668.564" "657.870" "681.081" "670.136" "699.228" "669.546" "669.928" "697.370"]	["614.213" "579.495" "547.322" "665.032" "526.139" "599.763" "595.060" "606.521" "580.290" "635.644"]	["575.502" "546.852" "548.705" "635.982" "642.375" "569.012" "614.898" "588.759" "555.027" "604.036"]	["665.639" "714.612" "664.491" "673.091" "644.185" "672.763" "687.474" "673.425" "692.723" "657.527"]
New Deltas	["0.001" "0.002" "0.002" "0.001" "0.001" "0.002" "0.002" "0.001" "0.001" "0.002"]	["0.003" "0.002" "0.003" "0.002" "0.001" "0.001" "0.001" "0.001" "0.003" "0.001"]	["0.003" "0.003" "0.003" "0.002" "0.003" "0.002" "0.003" "0.003" "0.003" "0.003"]	["0.003" "0.004" "0.002" "0.004" "0.003" "0.004" "0.004" "0.003" "0.002" "0.004"]	["0.001" "0.002" "0.001" "0.001" "0.001" "0.002" "0.001" "0.002" "0.002" "0.001"]
PD List	(["0.010" "0.017" "0.017" "0.017" "0.016" "0.019" "0.022" "0.017" "0.016" "0.017"])	(["0.046" "0.033" "0.025" "0.036" "0.023" "0.021" "0.021" "0.024" "0.038" "0.020"])	(["0.039" "0.040" "0.041" "0.033" "0.040" "0.037" "0.044" "0.043" "0.043" "0.045"])	(["0.041" "0.048" "0.036" "0.047" "0.038" "0.049" "0.049" "0.037" "0.035" "0.049"])	(["0.011" "0.020" "0.014" "0.016" "0.016" "0.018" "0.011" "0.019" "0.020" "0.018"])
Supply	(["1142.732" "1084.714" "1127.871" "1081.609" "1166.147" "1061.588" "1042.817" "992.215" "1094.084" "1137.224"])	(["897.642" "968.529" "1054.499" "847.697" "911.125" "1180.970" "912.879" "826.190" "981.035" "861.621"])	(["1000.000" "1000.000" "1000.000" "1000.000" "1000.000" "1000.000" "1000.000" "1000.000" "1000.000" "1000.000"])	(["1000.000" "1000.000" "1000.000" "1000.000" "1000.000" "1000.000" "1000.000" "1000.000" "1000.000" "1000.000"])	(["1139.636" "1069.750" "1147.076" "1123.707" "1179.476" "1136.246" "1102.217" "1134.450" "1107.502" "1152.633"])
Demand	(["1131.876" "1066.842" "1108.650" "1062.971" "1147.294" "1041.552" "1020.291" "975.771" "1076.266" "1117.662"])	(["857.593" "936.746" "1027.984" "817.520" "890.618" "1156.081" "893.613" "806.598" "944.043" "844.550"])	(["961.377" "961.175" "959.651" "967.647" "960.610" "964.104" "957.304" "957.525" "957.801" "956.049"])	(["959.674" "953.376" "964.154" "953.967" "962.628" "951.993" "952.012" "963.256" "966.055" "952.155"])	(["1127.279" "1048.543" "1131.295" "1105.425" "1160.802" "1115.572" "1090.572" "1113.397" "1085.732" "1132.502"])
Surplus	(["10.857" "17.872" "19.221" "18.639" "18.853" "20.035" "22.526" "16.444" "17.818" "19.562"])	(["40.049" "31.783" "26.515" "30.178" "20.507" "24.889" "19.265" "19.592" "36.991" "17.071"])	(["38.623" "38.825" "40.349" "32.353" "39.390" "35.896" "42.696" "42.475" "42.199" "43.951"])	(["40.326" "46.624" "35.846" "46.033" "37.372" "48.007" "47.988" "36.744" "33.945" "47.845"])	(["12.358" "21.207" "15.781" "18.282" "18.674" "20.673" "11.645" "21.053" "21.770" "20.131"])
Percent Surplus / Threshold Met?	(["0.955" "1.661" "1.719" "1.738" "1.630" "1.905" "2.184" "1.671" "1.642" "1.735"])	(["4.563" "3.336" "2.546" "3.624" "2.276" "2.130" "2.133" "2.400" "3.843" "2.001"])	(["3.938" "3.959" "4.118" "3.288" "4.018" "3.655" "4.363" "4.340" "4.311" "4.494"])	(["4.116" "4.774" "3.650" "4.712" "3.808" "4.919" "4.917" "3.743" "3.453" "4.902"])	(["1.090" "2.002" "1.385" "1.640" "1.596" "1.836" "1.062" "1.873" "1.985" "1.762"])
Percent Surplus Prev.	(["0.611" "2.447" "2.822" "2.304" "2.324" "3.518" "3.754" "1.659" "2.236" "2.475"])	(["4.079" "2.674" "3.738" "3.298" "1.187" "1.107" "1.160" "1.786" "3.530" "1.514"])	(["3.592" "3.385" "3.932" "2.960" "3.630" "2.909" "4.371" "4.045" "4.163" "4.286"])	(["3.582" "4.520" "3.080" "4.642" "3.330" "4.811" "4.939" "3.319" "2.734" "5.106"])	(["0.684" "3.596" "1.970" "1.840" "2.271" "2.694" "1.272" "3.114" "3.091" "2.326"])

Figure 2 – Implementation of the participatory planning procedure as a color-coded web app.

==

There are two other notable addenda to the pseudocode presented here that deserve mention. One, we simulated a two-year simulation of the participatory planning procedure. First, we simulated Year One of our participatory economy effectively from scratch as outlined in pseudocode above, then we preserved that information for a Year Two simulation with the same economy and implemented an "augmented reset" as follows:

1. Reset the iteration counter back to zero.
2. For each worker council, randomly increase the input-exponents, nature-exponents, and labor-exponents by a small amount (selected from the set [0, 0.001 0.002, 0.003, 0.004])
3. For each consumer council, randomly increase the private-good-exponents and public-good-exponents by a small amount (selected from the set [-0.002, -0.001, 0, 0.001 0.002]). Naturally, since we occasionally added negative numbers this will have the effect of reducing the size of the exponent in certain cases.
4. Repeat the procedure starting from step three of the above pseudocode.

The adjustments in the exponents simulate the improvements of technology in the economy during the passage of a given year.

To test the robustness of the procedure we intentionally violated our assumption requiring that the combined set of intermediate input-exponents, nature-exponents, and labor-exponents in a given worker's council always summed to less than one. The pseudocode for simulating increasing returns to scale is as follows:

1. In step one of the pseudocode above, randomly select 20% of the worker councils in the economy.
2. For that 20%, increase the size of each of the intermediate input-exponents, nature-exponents, and labor-exponents by 0.1, thus assuring that the total will be greater than one.
3. Proceed with the rest of the pseudocode as directed.

Results

The pseudocode was first implemented in the Netlogo environment, then was ported to the programming languages Clojure and Clojurescript. The transition to Clojure and Clojurescript made more feasible the expansion of the repertoire of types of goods and number of councils to better approximate the size of real world economies. In the largest scale considered, we generated forty experiments each comprising thirty thousand separate worker councils, thirty thousand consumer councils, and one hundred different goods. The results were first published in chapter nine of Hahnel (2021), and were highly encouraging.

When striving for a 5% threshold across two years of the lifespan of a simulated participatory planning procedure, we see an average of 6.5 iterations and an average GDP percentage increase of 2.446%. Even when we incorporate increasing returns to scale, the numbers are comparable: an average of 6.25 iterations, and an average GDP percentage increase of 1.859%. For the first time in democratic planning research, we have a prediction about its “practicality” backed up by *some* empirical support. The source code for all instances of the algorithm and pseudocode presented here are online[10], as are the data files for the large-scale experiments reported.[11]

Proposed Future Directions

There are a variety of proposed future directions for research involving pseudocode and algorithms for democratic-planning. These include the following:

1. *Other functions besides Cobb Douglas*. Cobb Douglas functions generate constant elasticities of demands for all goods, in contrast to real world economies where different goods can have very different elasticities. This algorithm can incorporate such functions by replacing our Cobb-

---

[10] The Clojure and Clojurescript code may be found at <https://github.com/msszczep/pequod-cljs>; a version of the original Netlogo instance of Pequod may be found at <https://github.com/msszczep/pequod2>

[11] The files are downloadable as gzipped Clojure data files at http://www.szcz.org/depexperiments

Douglas-powered model with one or more other functions, and then replacing the resulting production and utility functions in steps four and five in the pseudocode.

2. *Improved price adjustment algorithms*. The price adjustment rule we used in the algorithm and in the pseudocode was arrived at with little concerted effort. Despite the encouraging results we believe we can do better. We intend to carry out more systematic research across the price adjustment space to find more efficient price adjustment rules.

3. *Environmental impacts*. We have given precious little attention in this algorithm and in this research to the environmental impacts of a participatory economy. We hope and believe that a participatory economy would treat the environment with greater care than either its capitalist or communist predecessors -- an urgent question given the environmental calamities that await humanity if an environmentally-sound economy is not soon found. But hope and belief in a model or in a new system is not evidence. We should update the algorithm to incorporate pollutants, to examine how and if a participatory economy can minimize and reduce pollution (like the greenhouse gases that threaten runaway global climate change).

4. *Further robustness testing*. We added in some robustness testing in our pseudocode and in our testing (the increasing returns to scale), but more is needed to confirm that a participatory economy can handle different problems and concerns. We intend to update the algorithm and the pseudocode to violate different assumptions and test to see how the model handles those, if the economy convergences to a feasible allocation plan as quickly as before, more slowly, or not at all.

5. *Human intervention research*. The literature about a participatory economy offers the possibility of human intervention in the participatory planning procedure, particularly in later iterations where parts of the convergence may be slow or less-than-effective. This would help speed the arrival of an allocation plan. The pseudocode could be updated to see if such interventions prove helpful.

6. *More interactions among councils*. In all of the simulations thus far simulated in computer code, worker councils and consumer councils interact with the IFB, but the councils do not otherwise interact with each other. The model allows for federations of both worker councils and consumer councils to account for collective consumption at regional, national, and international levels, as well as changes in industry output by changing the number of worker councils in the industry rather than changing the outputs of existing councils. The pseudocode could be updated to allow for complex interactions among councils and more sophisticated council arrangements -- again, making for a still more complicated program and algorithm, but one more in tune to the proposal and vision of a participatory economy.

7. *Augment with other interactive software*. There is now a proposal to "design and build interactive participatory planning software in order to experiment with, improve upon and deepen our understanding of participatory planning in practice…and…in the longer-term, to build…planning software that could be used by a society wishing to implement a real-world pilot scheme." [12] Components of the research presented here could be utilized by this new software

[12] https://participatoryeconomy.org/project/participatory-planning-user-story-mapping/

project. Meanwhile, insights from this new project could help inform new research directions for this algorithm and accompanying pseudocode.

Conclusion

Alec Nove's claim that "there is no third way" beyond markets or command planning was an assertion unaccompanied by a logical argument. Our previous theoretical research describing how a participatory economy might function *in theory* was a first shot across the bow to Nove's assertion.

Now Nove's claim of "no third way" faces a new challenge: an implementation of the allocation mechanism for a democratically planned economy as a computer program. The algorithm of that program, outlined here as reasonably detailed pseudocode, and already implemented as working code, is available for examination, criticism, further testing and improvement.

The results so far seem to confirm that a "third way" economy is not only possible in theory, but seems to be a practical possibility as well. A much more compelling test of a democratically planned economy would be to implement one for some national economy in the real world. Absent that, we have pursued the next best option: implementing such an economy as a computer program, to address reasonable concerns about the economy's practicality and robustness.

"There was no third way" -- until there was!

References

Albert, Michael. *No Bosses.* (London: Zero Books, 2021)

Albert, Michael, and Hahnel, Robin. *The Political Economy of Participatory Economics.* (Princeton, New Jersey: Princeton University Press, 1991).

Hahnel, Robin. *Democratic Economic Planning.* (London: Routledge, 2021).

Hahnel, Robin. *A Participatory Economy.* (Chico, California: AK Press, 2022).

Hahnel, Robin, and Wright, Erik Olin. *Alternatives to Capitalism* (London: Verso Books, 2016).

Nove, Alec. *The Economics of Feasible Socialism* (London: Routledge, 1983).