SVVAMP: Simulator of Various Voting Algorithms in Manipulating Populations
François Durand, Fabien Mathieu, Ludovic Noirie

To cite this version:
François Durand, Fabien Mathieu, Ludovic Noirie. SVVAMP: Simulator of Various Voting Algorithms in Manipulating Populations. [Research Report] Inria. 2015. <hal-01135109>

HAL Id: hal-01135109
https://hal.archives-ouvertes.fr/hal-01135109
Submitted on 12 Dec 2015

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
SVVAMP: Simulator of Various Voting Algorithms in Manipulating Populations

François Durand
Paris Dauphine University

Fabien Mathieu and Ludovic Noirie
Alcatel-Lucent Bell Labs France

Abstract
We present SVVAMP, a Python package dedicated to the study of voting systems with an emphasis on manipulation analysis.

Introduction
History of voting theory has been marked by the discovery of several paradoxes, such as Gibbard–Satterthwaite impossibility theorem on manipulation (Gibbard 1973; Satterthwaite 1975). Since no reasonable voting system can avoid these paradoxes totally, their likeliness of occurrence under various probability assumptions or in real-life elections has been studied at length. However, there remain open questions in the domain, especially about the relative performance of various voting systems according to different criteria and under different sets of assumptions on the preferences of the voters.

Recently, interesting results were published about algorithmic issues linked to voting systems and their manipulation (Bartholdi and Orlin 1991; Xia et al. 2009; Walsh 2010; Zuckerman, Procaccia, and Rosenschein 2009; Zuckerman, Lev, and Rosenschein 2011; Gaspers et al. 2013). However, to the best of our knowledge, there was no publicly available software building on these existing techniques, in particular for the study of manipulability.

This observation led us to develop SVVAMP (Simulator of Various Voting Algorithms in Manipulating Populations), a Python package designed to study voting systems and their manipulability.

Voters’ preferences can be imported from external files or generated by a variety of probabilistic models. SVVAMP currently implements more than 20 voting systems, and its object-oriented design facilitates the implementation of new voting systems. Special attention has been paid to Coalitional Manipulability (CM) and its variants. Algorithms for Condorcet efficiency, Individual Manipulability (IM) and Independence of Irrelevant Alternatives (IIA) are also implemented.

Functionalities
SVVAMP can investigate multiple manipulation-related criteria for a large set of populations and voting systems.

Importing / creating populations
Populations in SVVAMP can be described through ordinal or cardinal preferences. Cardinal preferences are transparently converted to rankings whenever necessary. Importing a population from an external file is straightforward: SVVAMP can read simple CSV files containing the utilities of the population or files using the PrefLib format (Mattei and Walsh 2013). To generate artificial random populations, SVVAMP implements a variety of probabilistic models (cultures):

- Spheroid,
- Cubic Uniform and Ladder, three extensions of the Impartial Culture.
- Gaussian Well and Euclidean Box, two geometric models, which can be for instance used to produce single-peaked populations.
- Von Mises–Fisher, which is similar to Mallows’ model (Mallows 1957), but outputs cardinal preferences.

For any given population, SVVAMP can produce basic analysis: existence of a Condorcet winner, Borda and Plurality scores, …

Implemented Voting Systems

Studying Manipulability
For any given election (combination of a population and a voting system), SVVAMP can decide, in addition to the sincere winner $w$ of the election, the following issues:

- Independence of Irrelevant Alternatives (IIA): is $w$ still the winner when the election is held with any subset of the candidates including $w$? IIA is a central notion in Arrow’s celebrated impossibility theorem (Arrow 1950).
- Individual manipulation (IM): can a voter $v$, by casting an insincere ballot, secure an outcome $c$ that she strictly prefers to $w$ (while other voters still vote sincerely).
Coalitional manipulation: can a subset of voters, by cast-
ing insincere ballots, secure an outcome $c$ that they strictly
prefer to $w$ (while other voters still vote sincerely).

Ignorant-Coalition Manipulation (ICM), Unison-
Manipulation (UM) and Trivial Manipulation (TM), three
alternative types of coalitional manipulation.

Technical details

Algorithms

Determining manipulability, especially CM, can be computa-
tionally challenging (for example, it is NP-complete for
Borda Rule, Maximin, Coombs method and IRV).

$\text{SVAMP}$ is the first publicly available software imple-
menting state-of-the-art algorithms (Xia et al. 2009; Zuck-
erman, Procaccia, and Rosenschein 2009; Zuckerman, Lev,
and Rosenschein 2011; Gaspers et al. 2013; Walsh 2010)
and original heuristics. By default, it tries its most precise
algorithm among those running in polynomial time (exact
computation can be specified). Approximations convention-
ally return NaN if they cannot decide.

$\text{SVAMP}$ also embeds brute force algorithms to pro-
vide exact computation for any voting system (only recom-
mended for small instances).

Architecture

$\text{SVAMP}$ is written in a modular way. For instance, testing
CM is defined in class Election and calls a set of specific
sub-functions. Each of these sub-functions can be overrid-
den in the subclass implementing a specific voting system,
while keeping the others. This facilitates the definition of
new voting systems.

These generic methods defined in $\text{SVAMP}$ allow devel-
opers to quickly define a new voting system, only by its rule,
and already benefit from generic manipulation algorithms,
which makes $\text{SVAMP}$ easily extensible.

Voting systems also come with special attributes that rep-
resent a variety of properties that are used to avoid unnec-
eesary computations. For instance, if a voting systems verifies
the Condorcet criterium and if a population admits a Con-
dorcet winner, then $\text{SVAMP}$ immediately concludes that the
respective election meets the IIA criterium.

Also note that $\text{SVAMP}$ tries to be as lazy as possible. For
example, if asked to determine if an election is CM, it will
first perform some preliminary checks based on election’s
properties, then it will cycle through the candidate until it
finds a manipulation. If later one wants to get the list of
candidates for which a manipulation is possible, $\text{SVAMP}$
resumes the computation where it stopped.

Performance

$\text{SVAMP}$ is designed to run large scale experiments on reg-
ular computers. To give an order of magnitude, a full study
of all voting systems on 10,000 populations drawn with the
Spheroid culture, with $V = 33$ voters and $C = 5$ candidates
takes less than one half-hour on a 2.3 GHz personal laptop.

Available code

$\text{SVAMP}$ is a free software, under GNU General Public Li-
cense version 3. Its documentation includes installation pro-
ceedure, tutorials, reference guide and instructions for new
contributors. It is available at:

https://svvamp.readthedocs.org

We hope that it will be useful to researchers, teachers and
students interested in voting theory.

Acknowledgment

The work presented in this paper has been carried out at
$\text{LINCS}$ (http://www.lincs.fr).

References


354.

Gaspers, S.; Kalinowski, T.; Narodytska, N.; and Walsh,
T. 2013. Coalitional manipulation for Schulze’s rule. In
Proceedings of the 2013 international conference on Au-

Gibbard, A. 1973. Manipulation of voting schemes: A gen-

114–130.

Mattei, N., and Walsh, T. 2013. Preflib: A library of prefer-
ence data. In Proceedings of Third International Conference
on Algorithmic Decision Theory (ADT 2013).

Satterthwaite, M. 1975. Strategy-proofness and Arrow’s
conditions: Existence and correspondence theorems for vot-
ing procedures and social welfare functions. Journal of Eco-

Tideman, N. 2006. Collective Decisions And Voting: The
Potential for Public Choice. Ashgate.

In Computational Foundations of Social Choice, number
10101.

Xia, L.; Zuckerman, M.; Procaccia, A.; Conitzer, V.; and
Rosenschein, J. 2009. Complexity of unweighted coalitional
manipulation under some common voting rules. In Interna-
tional Joint Conference on Artificial Intelligence, 348–353.

Zuckerman, M.; Lev, O.; and Rosenschein, J. 2011. An algo-
rithm for the coalitional manipulation problem under Max-
imin. In The 10th International Conference on Autonomous

Algorithms for the coalitional manipulation problem. Artificial
Intelligence 173(2):392–412.