POLITECNICO DI MILANO

Facoltà di Ingegneria

Scuola di Ingegneria Industriale e dell'Informazione

Dipartimento di Elettronica, Informazione e Bioingegneria

Master of Science in

Computer Science and Engineering

Ingegneria Informatica

Stochastic Variance Reduced Policy Gradient

Supervisor:

MARCELLO RESTELLI

Assistant Supervisor:

MATTEO PAPINI

MATTEO PIROTTA

Master Graduation Thesis by:

DAMIANO BINAGHI
Student Id n. 858458

GIUSEPPE CANONACO
Student Id n. 852749

Academic Year 2017-2018

Final Version as of February 22, 2018 (classicthesis version 1.0).

Dediche

ACKNOWLEDGMENTS

Here you can put acknowledgements to people that helped you during the thesis. Remember that helping students to write thesis is part of the job of some of them, and they're also paid for that. Please make sure to thank them for what they weren't supposed to do.

Remember also that this page is part of your thesis. I know that your boyfriend/girlfriend is very important to you and you cannot live without her/him, as it is for me. But there's no need to put her/his name here unless she/he gave a proper contribution to this work. Same goes for friends, parents, drinking buddies and so on.

CONTENTS

Abstract ix
Estratto xi
Preface xiii
1 INTRODUCTION 1
2 STATE OF THE ART 3
2.1 Policy Gradient 3
2.2 Install LaTeX 4
2.2.1 Online editor 5
2.3 Use LaTeX within this template 6
2.3.1 File structure 6
2.3.2 Special environments 7
2.3.3 Citing, quoting and referencing 8
2.3.4 Figures and tables 8
2.3.5 Math 10
2.4 Contributing to this template 12
BIBLIOGRAPHY 13
A APPENDIX EXAMPLE: CODE LISTINGS 13
A.1 The listings package to include source code 13

LIST OF FIGURES

Figure 2.1 Thing taken from our master thesis 9

LIST OF TABLES

Table 2.1 Parameters needed for things 9

LISTINGS

Listing A.1 Code snippet with the recursive function to evaluate the pdf of the sum Z_N of N random variables equal to X. 14

ACRONYMS

SVRPG Stochastic Variance Reduced Policy Gradient

SVRG Stochastic Variance Reduced Gradient

MDP Markov Decision Processes

RL Reinforcement Learning

ABSTRACT

In this thesis, we propose a novel Reinforcement Learning (RL) algorithm consisting in a stochastic variance-reduced version of policy gradient for solving Markov Decision Processes (MDPs).
Stochastic variance-reduced gradient (SVRG) methods have proven to be very successful in supervised learning. However, their adaptation to policy gradient is not straightforward and needs to account for I) a non-concave objective function; II) approximations in the full gradient computation; and III) a non-stationary sampling process. The result is SVRPG, a stochastic variance reduced policy gradient algorithm that leverages on importance weights to preserve the unbiasedness of the gradient estimate. Under standard assumptions on the MDP, we provide convergence guarantees for SVRPG with a convergence rate that is linear under increasing batch sizes. Finally, we suggest practical variants of SVRPG, and we empirically evaluate them on continuous MDPs.

SOMMARIO

In questa tesi proponiamo un nuovo algoritmo nell'ambito del Reinforcement Learning (RL). Questo algoritmo consiste nella versione di stochastic variance-reduced del gradiente della politica per la risoluzione dei processi decisionali di Markov (MDPs).
Il metodo di riduzione della varianza del gradiente, chiamato Stochastic variance-reduced gradient (SVRG), ha avuto molto successo nell'ambito dell'apprendimento supervisionato. La sua adattazione nel contesto del gradiente della politica non banale e necessita di alcune considerazioni: I) la funzione obiettivo non concava; II) il full gradient deve essere necessariamente approssimato; III) il processo di campionamento non stazionario. Il risultato SVRPG, un algoritmo di riduzione della varianza del gradiente della politica che sfrutta gli importance weights per preservare la correttezza della stima del gradiente. Date le classiche assunzioni del MDP, abbiamo fornito garanzie di convergenza per SVRPG con un tasso di convergenza che lineare al crescere della dimensione del batch. Infine abbiamo implementato una variante pratica di SVRPG. Abbiamo valutato questa variante empiricamente su dei MDP cintinui.

ESTRATTO

"... il testo delle tesi redatte in lingua straniera dovr essere introdotto da un ampio estratto in lingua italiana, che andr collocato dopo labstract."

PREFACE

A preface is an introduction to a book or other literary work written by the work's author. A preface generally covers the story of how the book came into being, or how the idea for the book was developed.

MOTIVATION

Graduating is not the motivation that one expects here.

1 INTRODUCTION

Reinforcement Learning Reinforcement Learning (RL) is a field of machine learning that aims at building intelligent machines, called agent, capable of learning complex tasks from experience. The goal of RL [**sutton1998reinforcement**] algorithm is to learn the best actions by direct interaction with the environment and evaluation of the performance in the form of a reward signal.

2 STATE OF THE ART

In this section, we provide the essential background on policy gradient methods and stochastic variance-reduced gradient methods for finite-sum optimization.

2.1 POLICY GRADIENT

A Reinforcement Learning task [**sutton1998reinforcement**] can be modelled with a discrete-time continuous Markov Decision Process (MDP) $M = \{\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma, \rho\}$, where $\mathcal{S}$ is a continuous state space; $\mathcal{A}$ is a continuous action space; $\mathcal{P}$ is a Markovian transition model, where $\mathcal{P}(s'|s,a)$ defines the transition density from state s to s' under action a; $\mathcal{R}$ is the reward function, where $\mathcal{R}(s,a) \in [-R, R]$ is the expected reward for state-action pair (s,a); $\gamma \in [0,1)$ is the discount factor; and ρ is the initial state distribution. The agent's behaviour is modelled as a policy π, where $\pi(\cdot|s)$ is the density distribution over $\mathcal{A}$ in state s. We consider episodic MDPs with effective horizon H.[1] In this setting, we can limit our attention to trajectories of length H. A trajectory τ is a sequence of states and actions $(s_0, a_0, s_1, a_1, \dots, s_{H-1}, a_{H-1})$ observed by following a stationary policy, where $s_0 \sim \rho$. We denote with $p(\tau|\pi)$ the density distribution induced by policy π on the set $\mathcal{T}$ of all possible trajectories (see Appendix **??** for the definition), and with $\mathcal{R}(\tau)$ the total discounted reward provided by trajectory τ: $\mathcal{R}(\tau) = \sum_{t=0}^{H-1} \gamma^t \mathcal{R}(s_t, a_t)$. Policies can be ranked based on their expected total reward: $J(\pi) = \mathbb{E}_{\tau \sim p(\cdot|\pi)} [\mathcal{R}(\tau)|M]$. Solving an MDP M means finding $\pi^* \in \arg\max_{\pi}\{J(\pi)\}$.

Policy gradient methods restrict the search for the best performing policy over a class of parametrized policies $\Pi_\theta = \{\pi_\theta : \theta \in \mathbb{R}^d\}$, with the only constraint that π_θ is differentiable w.r.t. θ. For sake of brevity, we will denote the performance of a parametric policy with $J(\theta)$ and the probability of a trajectory τ with $p(\tau|\theta)$ (in some occasions, $p(\tau|\theta)$ will be replaced by $p_\theta(\tau)$ for the sake of readability). The search for a locally optimal policy is performed through gradient ascent, where the policy gradient is [**sutton2000policy**, **Peters2008reinf**]:

$$\nabla J(\theta) = \mathop{\mathbb{E}}_{\tau \sim p(\cdot|\theta)} \left[\nabla \log p_\theta(\tau) \mathcal{R}(\tau)\right]. \tag{2.1}$$

Notice that the distribution defining the gradient is induced by the current policy. This aspect introduces a nonstationarity in the sampling process. Since the underlying distribution changes over time, it

1 The episode duration is a random variable, but the optimal policy can reach the target state (i.e., absorbing state) in at most H steps. This has not to be confused with a finite horizon problem where the optimal policy is non-stationary.

is necessary to resample at each update or use weighting techniques such as importance sampling. Here, we consider the *online learning scenario*, where trajectories are sampled by interacting with the environment at each policy change. In this setting, stochastic gradient ascent is typically employed. At each iteration $k > 0$, a batch $\mathcal{D}_N^k = \{\tau_i\}_{i=0}^N$ of $N > 0$ trajectories is collected using policy π_{θ_k}. The policy is then updated as $\theta_{k+1} = \theta_k + \alpha \widehat{\nabla}_N J(\theta_k)$, where α is a step size and $\widehat{\nabla}_N J(\theta)$ is an estimate of Eq. (2.1) using $\mathcal{D}_N^k$. The most common policy gradient estimators (e.g., REINFORCE [**williams1992simple**] and G(PO)MDP [**baxter2001infinite**]) can be expressed as follows

$$\widehat{\nabla}_N J(\theta) = \frac{1}{N} \sum_{n=1}^{N} g(\tau_i|\theta), \quad \tau_i \in \mathcal{D}_N^k, \tag{2.2}$$

where $g(\tau_i|\theta)$ is an estimate of $\nabla \log p_\theta(\tau_i)\mathcal{R}(\tau_i)$. Although the REINFORCE definition is simpler than the G(PO)MDP one, the latter is usually preferred due to its lower variance. We refer the reader to Appendix **??** for details and a formal definition of g.

The main limitation of plain policy gradient is the high variance of these estimators. The naïve approach of increasing the batch size is not an option in RL due to the high cost of collecting samples, i.e., by interacting with the environment. For this reason, literature has focused on the introduction of baselines (i.e., functions $b : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$) aiming to reduce the variance [**williams1992simple**, **Peters2008reinf**, **Thomas2017actionbaseline**, **wu2018variance**], see Appendix **??** for a formal definition of b. These baselines are usually designed to minimize the variance of the gradient estimate, but even them need to be estimated from data, partially reducing their effectiveness. On the other hand, there has been a surge of recent interest in variance reduction techniques for gradient optimization in supervised learning (SL). Although these techniques have been mainly derived for finite-sum problems, we will show in Section **??** how they can be used in RL. In particular, we will show that the proposed SVRPG algorithm can take the best of both worlds (i.e., SL and RL) since it can be plugged into a policy gradient estimate using baselines. The next section has the aim to describe variance reduction techniques for finite-sum problems. In particular, we will present the SVRG algorithm that is at the core of this work.

2.2 INSTALL LaTeX

If you don't have already a LaTeX system installed, this section will explain everything you need. The easiest way to get LaTeX is to install TeXLive, which works on all **OS!**s (OS!s). In https://www.tug.org/texlive/ you find the instructions and the files needed - and also get in touch with minimalism of TEXusers.

Then you will need an editor: I strongly recommend TeXworks because it's very simple and available on all the platforms. Also you don't need to install it, it's already included in TeXLive. The official documentation of TeXworks is available here;[2] I strongly recommend the reading of chapter 3. Alternatevely you can read an italian manual: profs.sci.univr.it/ gregorio/introtexworks.pdf (just 13 pages, read it!).[3]

After opening TeXworks, I strongly suggest to set these two additional things:

- open Preferences, then go the Composition tab: in the second box there, the "Process instruments", push the plus button. In the window just opened, write Biber in the "Name" field, biber in the "Program" field (lowercase!) and then press the plus button to add the argument $basename;
- again in the same window, set "Hide console output" to "never".

Then just test the installation of the template:

a. go into the template home folder;

b. open the file ClassicThesis˙DEIB.tex;

c. select pdfLaTeX from the dropdown menu in the top right of the TeXworks window;

d. press the rounded green button: it compiles the .tex file for the first time and open the resulting .pdf;

e. select Biber from the same dropdown menu and press again the green button: this compiles the bibliography, a thing you need to repeat only when you change the file Bibliography.bib;

f. select pdfLaTeX again and recompile: this is needed to build indices and crossreferences;

The above compilation procedure is the standard way to translate the LaTeX code into pdfs.

2.2.1 *Online editor*

If the above procedure seems too difficult to you and you have an internet connection always available, you might think to use an online editor. The best choice at the time of writing is http:\\sharelatex.com where you can even find this template after registration to the site by looking for "Classic Thesis At DEIB". Your project will be saved on

2 https://docs.google.com/file/d/0B5iVT8Q7W44pMk1WSFRKcDRlMU0/preview
3 If you already have a preferred editor, just keep using yours.

their server but you can also download them. The platform allows up to two authors for free accounts.

There is no need to provide instructions for its use since the website has them. They also have an online LaTeXguide.

2.3 USE LaTeX WITHIN THIS TEMPLATE

2.3.1 *File structure*

The template is organized in multiple file and folders:

A. ClassicThesis˙DEIB.tex is the main file to be compiled, found in the root folder. You should just add the source filenames you want to include and any hyphenation you need to explictly specify.

B. classicthesis-config.tex contains options that can be chosen for this template, like the draft one that prints date and time at the bottom of every page. It contains also the definition for the title, the author and others stuff displayed in the titlepage. Comments within the file should guide you.[4] Take a look at it!

C. Bibliography.bib is the *Bibtex* database: it is a normal textfile where you should put books and articles read;

D. Chapters contains the files for the main chapters of your thesis; this is where you will add the chapters text, as well these very words in line 41 of the file Conclusion.tex;

E. CodeFiles contains any code snippet you want to include in your thesis with the environment listings; it might be some relevant Matlab or C code, as well as long bash scripts;

F. FrontBackmatter contains various files that are included in the main one to produce abstract, titlepages, acknowledgements, ... Follow the instructions below to modify them in order to suits your needs;

G. Images contains the .pdf or .png versions of the images of the thesis organized in subfolder per chapter.

To modify abstract, preface, acknowledgements snd acronyms, you need to go into the folder FrontBackmatter where you will find the following:

ABSTRACT.TEX contains the text displayed as “abstract” and “sommario” just after the list of figures, tables, etc. Modify the text and leave the rest.

4 comments are the rows starting with %.

ACKNOWLEDGMENTS.TEX contains the text put just before the table of contents. Modify the text to suit your needs.

ACRONYMS.TEX contains the environment acronym with the definition of all the acronyms that will be used within the text. Add your own to the list and put the longest as parameter of the environment.

AUTOPARTS folder contains things that should work without your intervention. Forget them.

DEDICATION.TEX same usage and structure as Acknowledgements.tex.

ESTRATTO.TEX Politecnico di Milano requires an italian long excerpt of theses written in foreign languages.

FRONTESPIZIO.TEX and FrontespizioIT.tex are the cover page in english and italian, respectively. Politecnico di Milano requires the italian version of the english cover, so there it is. Both should work perfectly if you modify section 2 of the file classicthesis-config.tex, but you may not like the style so modify them as you prefer.

PREFACE.TEX same usage and structure as Acknowledgements.tex.

PUBLICATION.TEX same usage and structure as Acknowledgements.tex, but not included by default. Activate it by uncommenting the relevant line in ClassicThesis˙DEIB.tex.

RETROFRONTESPIZIO.TEX contains the colophon. In most cases is fine as it already is.

2.3.2 *Special environments*

In addition to common LATEX environments, this thesis is set to use:

Use these environments: they make the thesis less bland and readable.

- the command "graffito–" is used to create margin notes. The limits in number of words or length of word must be seen as a motivation to keep the notes short and simple;
- "begin–aenumerate" to produce an "enumerate with letters instead of numbers, as in the file list above;
- footnotes are useful to provide extra information. Usually they are not required to understand a paragraph but provide interesting details. This keeps the main body of text concise. You can create them with "footnote–text".[5]

5 They should be placed after the punctuation mark and preferably at the end of the paragraph. In fact, they should not interrupt the reading flow. If you need to put a footnote in the middle of a paragraph, or of a sentence then the note should be part of the main text.

- “ac–” and its variations, defined by package acronyms, provide nice handling for acronyms, like **XML!** (**XML!**), produced with the code “ac–XML”. List them within the environment acronym in the file FrontBackmatter/Acronyms.tex.

2.3.3 *Citing, quoting and referencing*

References to bibliography are produced in the usual way with “cite–bib˙key” ([**bringhurst:2002**]); don’t forget the brackets which have to be added by hand. There also variations of the command, like “citeauthor–bib˙key”, “citetitle–bib˙key” and others that you can find in the bibtex manual.

“blockcquote[][]–”–” “produce a quotation with reference to author and page” [**bringhurst:2002**]. If the quotation is longer than two rows is indented. This behavior is provided by the package csquotes, which settings are in classicthesis-config.tex. The package also provides “enquote–the citation” that produces “correct quotation style” according to the language in use.

There is a set of commands to refer to chapters, sections, subsections, appendices, figures, tables and equations, like “myChap–label˙key” to produce chapter 1. There are also capital versions of the commands (“MyChap–” produces Chapter 1). They need a “label–name” anchor next to the referred thing.

- “myChap for chapters;
- “mySec for sections;
- “mySubsec for subsections;
- “myAppendix for appendices;
- “myFig for figures;
- “myTab for tables;
- “myEq for equations;

2.3.4 *Figures and tables*

Figures are handled usually with the code

“begin–figure”
“centering
“includegraphics[width=“columnwidth]–Images/your˙image˙name.pdf”
“caption[Short description]–Long description.”
“label–fig:a˙name”
“end–figure”

which produces things like figure 2.1. Take care of the short description: it appears in the list of figures and should be just a reference,

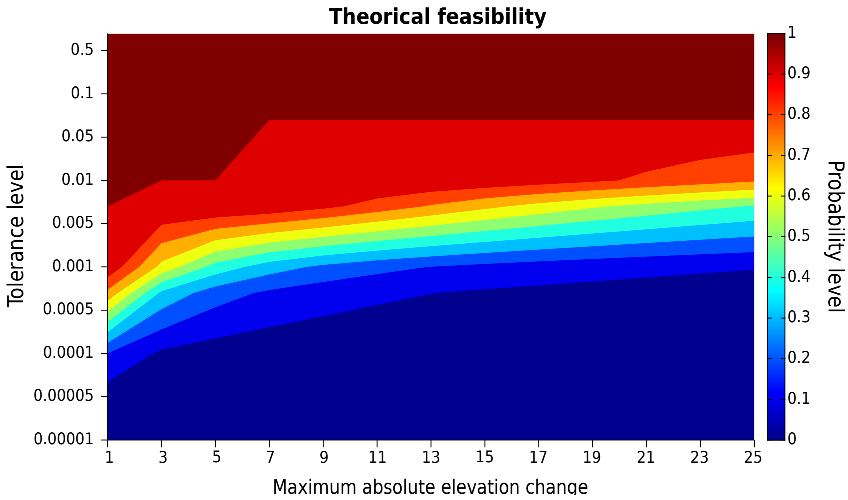

Figure 2.1: Thing taken from our master thesis whose meaning have been completely forgotten.

ALGORITHM	PARAMETER	SUGGESTED VALUES		
Any	NFE	10 000	÷	200 000
	Population Size	10	÷	1000
GDE3	DE step size	0.0	÷	1.0
	Crossover rate	0.0	÷	1.0

Table 2.1: Parameters needed for things that are not needed anymore themselves.

not a exhaustive description. Of course, you need to put the image file your˙image˙name.pdf in folder Images/. We suggest to keep things organized: create a folder for each chapter and keep the original source/working file in the same place.

Tables are produced with the code

```
“begin–table”[tb]
“footnotesize
“centering
“begin–tabularx”–0.8“textwidth”–llrcl”
“toprule
“tableheadline–l”–Algorithm” &
“tableheadline–l”–Parameter” &
“tableheadlineMore–3”–c”–Suggested Values” ““
“midrule
“tablefirstcol–l”–Any”
& “acs–NFE” & $10“,000 $ & $ “div $ & $ 200“,000$ ““
& Population Size & $10 $ & $ “div $ & $ 1000$ ““
“midrule
“tablefirstcol–l”–“ac–GDE3””
```

& “ac–DE” step size & $0.0 $ & $“div $ & $ 1.0$ ““
& Crossover rate & 0.0 & $ “div $ & $ 1.0$ ““
“bottomrule
“end–tabularx”
“caption[Short description]–Long description.”
“label–tab:MOEAandParameters”
“end–table”

which produces table 2.1. “myfloatalign, “tableheadline–”–” and its variation “tableheadlineMore–”–”–” and “tablefirstcol–”–” are used to give a common style to all tables in the document. Use them! They are defined in classicthesis-config.tex.

2.3.5 *Math*

You can produce an equation like $\lim_{n\to\infty}\sum_{k=1}^{n}\frac{1}{k^2} = \frac{\pi^2}{6}$ by embedding this code in the line:

$“lim˙–n “to “infty”“sum˙–k=1”ˆn “frac–1”–kˆ2”= “frac–“piˆ2”–6”$.

Equation that spans the full line like:

$$\lim_{n\to\infty}\sum_{k=1}^{n}\frac{1}{k^2} = \frac{\pi^2}{6}$$

are produced with something like this:

“[
“lim˙–n “to “infty”“sum˙–k=1”ˆn “frac–1”–kˆ2”= “frac–“piˆ2”–6”.
“]

If you need to refer to the equation later on, you need to number and label it. It is done via

“begin–equation”
“label–eq:euler”
eˆ–i“pi”+1=0.
“end–equation”

$$e^{i\pi} + 1 = 0. \tag{2.3}$$

From equation (2.3) you can see how “myEq–eq:euler” should be used.

Numeric sets requires specific font as $\forall x \in \mathbb{R}$ which is produced with $“forall x “in “mathbb–R”$. Matrices like

$$A = \begin{bmatrix} x_{11} & x_{12} & \dots \\ x_{21} & x_{22} & \dots \\ \vdots & \vdots & \ddots \end{bmatrix}$$

requires

A=
“begin–bmatrix”
x˙–11” & x˙–12” & “dots ““
x˙–21” & x˙–22” & “dots ““
“vdots & “vdots & “ddots
“end–bmatrix”.

Multiline equation can be produced with different environments like split and cases.

$$\begin{split} a &= b+c-d \\ &= e-f \\ &= g+h \\ &= i. \end{split}$$

comes from

“begin–split”
a &= b+c-d ““
&= e-f ““
&= g+h ““
&= i.
“end–split”.

$$f(n) := \begin{cases} 2n+1, & \text{con } n \text{ dispari,} \\ n/2, & \text{con } n \text{ pari.} \end{cases}$$

comes from

“begin–split”
a &= b+c-d ““
&= e-f ““
&= g+h ““
&= i.
“end–split”.

Definition like

Definition 2.1 (Gauss). The math guy find obvious that $\int_{-\infty}^{+\infty} e^{-x^2}\,dx = \sqrt{\pi}$.

are produced with the code

“begin–definition”[Gauss]
The math guy find obvious that
$“int˙–-“infty”ˆ–+“infty”
eˆ–-xˆ2”“,dx=“sqrt–“pi”$.
“end–definition”

There also a number of other similar environments, like observation, theorem with or without name, corollary and lemma.

Observation 2.2. *But many people like me don't find it obvious.*

Theorem 2.1. Mathematicians are very rare, if any.

Theorem 2.2 (Pythagorean)**.** The square of the hypotenuse of a triangle is equal to the sum of the squares of the other two sides.

Demonstration is left for exercise.

Corollary 2.3. *A line segment whose length is incommensurable so the ratio of which is not a rational number, can be constructed using a straightedge and compass.*

Lemma 2.4. *Pythagoras's theorem enables construction of incommensurable lengths because the hypotenuse of a triangle is related to the sides by the square root operation.*

You can also proof your theorem with the environment proof.

Theorem 2.5 (Surprise)**.** We have $\log(-1)^2 = 2\log(-1)$.

Proof. We have $\log(1)^2 = 2\log(1)$. But also we have $\log(-1)^2 = \log(1) = 0$. So $2\log(-1) = 0$. ■

There's also the cute little square at the end.

2.4 CONTRIBUTING TO THIS TEMPLATE

Suggestion and improvements are welcome at https://github.com/Lordmzn/ClassicThesis-at-DEIB or via email at emanuele.mason@polimi.it, andrea.cominola@polimi.it or daniela.anghileri@polimi.it.

A
APPENDIX EXAMPLE: CODE LISTINGS

We have seen that computer programming is an art,
because it applies accumulated knowledge to the world,
because it requires skill and ingenuity, and especially
because it produces objects of beauty.

— **knuth:1974**, **knuth:1974**, **knuth:1974**

A.1 THE listings PACKAGE TO INCLUDE SOURCE CODE

Source code is usually not part of the text of a thesis, but if it is an original contribution it makes sense to le the code speak by itself instead of describing it. The package listings provide the proper layout tools. Refer to its manual if you need to use it, an example is given in listing A.1.

Listing A.1: Code snippet with the recursive function to evaluate the pdf of the sum Z_N of N random variables equal to X.

```
std::vector¡int¿ values˙of˙x(number˙of˙values˙of˙x,
  min˙value˙of˙x);
for (unsigned int i = 1; i ¡ number˙of˙values˙of˙x; i
   ++) –
  values˙of˙x[i] = values˙of˙x[i - 1] + 1;
”

prob˙x = 1.0 / number˙of˙values˙of˙x;
std::vector¡std::vector¡double¿ ¿ p˙z;
for (unsigned int idx = 0; idx ¡ p˙z.size(); idx++) –
  p˙z[idx] = std::vector¡double¿(
    (max˙value˙of˙x * (idx + 1) - min˙value˙of˙x
      * (idx + 1)) + 1, INIT˙VALUE);
”


double prob(int Z, int value˙of˙z) –
  if (value˙of˙z ¡ min˙value˙of˙x * Z ——
    value˙of˙z ¿ max˙value˙of˙x * Z) –
      return 0.0;
  ”
  if (value˙of˙z ¡ min˙value˙of˙z ——
    value˙of˙z ¿ max˙value˙of˙z) –
      return 0.0;
  ”
  int idx˙value˙of˙z = -(min˙value˙of˙z - value˙of˙z);
  int idx˙N = Z - 1;
  if (p˙z[idx˙N][idx˙value˙of˙z] == -2.0) –
    if (Z ¿ 1) –
      double pp = 0.0;
      for (unsigned int i = 0; i ¡
         number˙of˙values˙of˙x; i++) –
        pp += prob(Z - 1, value˙of˙z - values˙of˙x[i],
           p);
      ”
      p˙z[idx˙N][idx˙value˙of˙z] = prob˙x * pp;
    ” else –
      if (Z == 1) –
        for (unsigned int j = 0; j ¡
           number˙of˙values˙of˙x; j++) –
          if (value˙of˙z == values˙of˙x[j]) –
            p˙z[idx˙N][idx˙value˙of˙z] = prob˙x;
            break;
          ”
        ”
      ”
      if (p˙z[idx˙N][idx˙value˙of˙z] == INIT˙VALUE) –
        p˙z[idx˙N][idx˙value˙of˙z] = 0.0;
      ”
    ”
  ”
  return p˙z[idx˙N][idx˙value˙of˙z];
”
```