

# **pipeline: TODO**

John Letey<sup>1</sup>, Tony E. Wong<sup>1</sup>

## **ABSTRACT**

TODO!!! The code is available online at <http://mussles.github.io/pipeline/> under the GNU General Public License v3.

*Subject headings:* methods: statistical — methods: Markov chain Monte Carlo  
— TODO

---

<sup>1</sup>University of Colorado at Boulder

*Note: If you want to get started immediately with the `pipeline` package, start at Appendix A on page 4 or visit the online documentation at <https://mussles.github.io/pipeline>. If you are sampling with `pipeline` and having low-acceptance-rate or other issues, there is some advice in Section 3 starting on page 3.*

## 1. Introduction

Throughout previous years of development, Markov chain Monte Carlo (MCMC) has evolved from a foregone algorithm to a more general way to fit real-life data.

## 2. The Algorithm

One of the most commonly used MCMC algorithms is the Adaptive Metropolis-Hastings Algorithm, proposed by Haario, et al. in their *An Adaptive Metropolis Algorithm* [Haario, et al. \(2001\)](#). In their paper, they propose an algorithm that streamlines the task of setting the step-size, compared to the arduous process outlined in the original algorithm. The original algorithm was introduced by Metropolis and Hastings, respectively, in their separate papers [Metropolis, et al. \(1953\)](#) and [Hastings \(1970\)](#). The algorithm that does TODO, was originally proposed by Metropolis, and was further refined by Hastings to include TODO. Contained in the following algorithms, is rough pseudo-code for both the Metropolis-Hastings algorithm and also the Adaptive Metropolis-Hastings Algorithm:

---

### Algorithm 1 The Metropolis-Hastings algorithm

---

- 1: Draw  $X^0 \sim \mu$  where  $\mu$  is the initial condition.
  - 2: Set  $t \leftarrow 0$ .
  - 3: **repeat**
  - 4:   Draw  $Y^t \sim q(y^t|x^t)$ .
  - 5:   Compute  $\alpha = \min \left\{ \frac{\pi(y^t) q(x^t|y^t)}{\pi(x^t) q(y^t|x^t)}, 1 \right\}$
  - 6:   Draw  $U \sim \mathcal{U}[0, 1]$ .
  - 7:   **if**  $U \leq \alpha$  **then**
  - 8:     Set  $X^{t+1} \leftarrow Y^t$ .
  - 9:   **else**
  - 10:     Set  $X^{t+1} \leftarrow X^t$ .
  - 11:   **end if**
  - 12:   Set  $t \leftarrow t + 1$ .
  - 13: **until** Sufficiently many samples have been produced.
-

---

**Algorithm 2** Adaptive Metropolis-Hastings algorithm

---

Draw  $X^0 \sim \mu$  where  $\mu$  is the initial condition.

Set  $\theta^0$  to an arbitrary valid value.

Set  $t \leftarrow 0$ .

**repeat**

    Compute  $\theta^t = \gamma^t(\theta^0, X^0, \dots, X^{t-1})$  where  $\gamma^t$  is a transformation that update the parameters based on past samples.

    Draw  $X^{t+1}$  using the proposal  $q(\cdot|x^t, \theta^t)$  with the Metropolis-Hastings rule.

    Set  $t \leftarrow t + 1$ .

**until** The accept rate of the last  $K$  iterations are close to 0.234.

Set  $\theta \leftarrow \theta^t$ .

Perform Algorithm 1 with proposal  $q(\cdot|x) = q(\cdot|x, \theta)$ .

---

### 3. Discussion & Tips

#### REFERENCES

- Haario, H.; Saksman, E.; Tamminen, J. (2001). “An adaptive Metropolis algorithm.” *Bernoulli*, 7(2): 223242.
- Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A., and Teller, E. (1953). “Equations of state calculations by fast computing machines.” *J. Chem. Phys.*, 21(6): 10871092.
- Hastings, W. (1970). “Monte Carlo sampling methods using Markov chains and their application.” *Biometrika*, 57: 97109.

## A. Installation

## B. Issues & Contributions

The development of `pipeline` is being coordinated on GitHub at <http://github.com/mussles/pipeline> and contributions are welcome. If you encounter any problems with the code, please report them at <http://github.com/mussles/pipeline/issues> and consider contributing a patch.

## C. Online Documentation

To learn more about how to use `pipeline` in practice, it is best to check out the documentation on the website <https://mussles.github.io/pipeline>. This page includes the API documentation and many examples of possible work flows.