Markov Chain Monte Carlo with People

Andrew Ang
CS 186 Final project

May 9, 2015

1 Introduction

Sanborn and Griffiths [1] describe a way to sample from subjective category distributions by designing a task where people act as elements of a MCMC algorithm. This represents an interesting combination of both human computation and information elicitation. We demonstrate how this method can be used to create representations of peoples' beliefs on color hues.

1.1 Markov Chain Monte Carlo

Markov Chain Monte Carlo (MCMC) algorithms are a collection of methods that enamble sampling from arbitrary distributions, by constructing a Markov chain such that its stationary distribution on states converges to the desired target distribution. The Metropolis-Hastings algorithm is one such MCMC method. Metropolis-Hastings uses a proposal function to propose new states of the chain, and an acceptance function to determine the probability of accepting the proposal. A basic outline of this method can be described as follows:

1. Initialize x_t from an initial distribution
2. Draw proposal x' from the proposal distribution q
3. Accept proposal with probability determined by the acceptance function $A(x', x_t)$
4. If proposal accepted, $x_t = x'$
5. Go to step 2 and repeat

After allowing enough iterations for convergence, the states x_t become samples from the target distribution.

The proposal distribution and acceptance function must be chosen carefully in order for the constructed Markov chain to converge to the target distribution $p(x)$. The proposal distribution must be symmetric ($q(x'|x_t) = q(x_t|x')$ where $q(x'|x_t)$ is the probability of x' given current state x_t). While there are many acceptance functions that are valid, but one relevant valid acceptance function is the Barker acceptance function,

$$A(x'; x_t) = \frac{p(x')}{p(x_t) + p(x')} \tag{1}$$

where p(x) is the probability of x in the target distribution.

1.2 MCMC with people

When people are given a choice between two items x_1 and x_2, and told to pick the item that better represents a particular category c, a phenomenon called probability matching is usually observed. Instead of maximizing the probability of a correct decision by choosing whichever item is more likely with probability 1, experimental evidence demonstrates that subjects tend to choose according to the Luce choice rule (ratio rule):

$$a(x_1, x_2) = \frac{p(x_1|c)}{p(x_1|c) + p(x_2|c)} \tag{2}$$

where $a(x_1, x_2)$ is the probability of choosing x_1 over x_2, and $p(x_1|c)$ is the probability of seeing x_1 given category c.

[1] points out that the choice rule is analogous to the Barker acceptance function, with the target probability in equation (1) being equivalent to the category distribution $p(x|c)$ in equation (2). This observation has two very interesting implications for human computation and information elicitation.

1.2.1 Human computation

Because people have a natural ability to easily calculate the acceptance function for target distribution $p(x)$ equal to some categorical distribution, it is possible to "run" MCMC using a human computation scheme. By asking subjects to make repeated binary choices and adapting the presentation of stimuli based on the previous choice and a valid proposal function, the Metropolis-Hastings algorithm with a Barker acceptance function can be recreated.

With any method involvin human computation, human error is an issue to consider. The Markov chain may not converge properly if subjects make enough mistakes on the binary choice task. We simulate the effect of error in accepting and rejecting proposals in our MCMC simulations.

1.2.2 Information elicitation

The target distribution used to calculate the analogous Barker acceptance rule (and ultimately the distribution being sampled by the MCMC algorithm) may be of interest, and useful to sample from. $p(x)$, the category distribution on items x, is a mental representation of a category. In [1], Sanborn and Griffiths apply MCMC with people to uncover mental representations of animals using stick figures of varying sizes and component lengths as MCMC stimuli. Any kind of mental representation likely takes into account many parameters, and this method is useful for creating representations of those distributions over high-parameter/high-dimensional spaces.

It might be interesting to design incentives such that information elicitation can occur via this MCMC method. In its current state, there is no strong incentive to provide truthful answers. One potential issue in incentivizing truthfulness is that because this method is dependent on a tendency to probability match, introducing a monetary incentive based on number of correct (assuming we know what is correct) answers may distort the calculation of the acceptance function. An input agreement mechanism based on degree of similarity between the realized distributions may be a possible strategy here.

1.3 Color Task

An interesting set of distributions that could be sampled with this MCMC method might be people's beliefs on specific color hues. The term "gray" may encompass many possible shades of gray, represented as a distribution over the possible color values. Likely some colors may be clearly defined (e.g. blue, white, red), while the distributions of others (e.g. sky blue, periwinkle, violet vs. purple) may be more subjective and varied.

The binary choice task used for MCMC in this context can be structured as shown in Figure 1: Given two shades of color (displayed as colors), which one is closer to a target color (displayed as text)? The target color represents the target color distribution we want to sample from. To parameterize colors we use the RGB color model to represent colors.

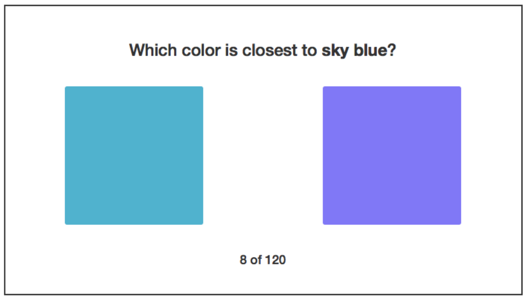


Figure 1: Color task interface

1.3.1 RGB color model

The RGB (Red-Green-Blue) color model is widely used for producing colors in computer monitors, televisions, and theaters. Colors are produced by adding together 3 components: red, green, and blue light. Different colors are produced depending on the amounts of each component used. While this system can not represent all visible colors, it does represent a very wide spectrum.

We can represent a value rgb(R,G,B) in the RGB color space with 3 parameters, $\{R, G, B\}$. A common color scheme used in computers and web browsers lets R, G, and B take integer values from $[0, 255]$.

1.3.2 Representing color distributions

To represent an explicit color distribution in the RGB space for the purposes of simulation, a beta distribution with support [0,256) is chosen to independently represent the R/G/B components. Using beta is convenient since it is defined over a bounded support.

2 Methods

2.1 Simulation of MCMC color sampling

A Python scipt was used to simulate MCMC sampling on the RGB color space using a repeated color choice.

To imitate the conditions used in subsequent mTurk studies, we simulate 50 separate MCMC instances, and we limit the number of proposals performed per chain to 120. We throw away MCMC samples produced before the 30th proposal to account for the "burn-in" period before convergence.

The initial sample x_t is chosen uniformly over the color space. The proposal distribution is multivariate normal, centered at the current MCMC state x_t with diagonal covariance matrix (diagonal values = 50). If any RGB component value in x_t is outside range [0,255], we resample from the proposal distribution until a valid proposal sample is found. (Validity of this methodology is explained in [2]; while a symmetric proposal distribution is required, here we can throw away samples outside the parameter range because they would be rejected anyways.) All samples x_t are rounded to integer values.

To attempt to simulate the effect of human error on accepting proper proposals, we also run a simulation where there is a 10% chance of rejecting the proposal if it was supposed to be accepted and 10% chance of accepting the proposal if it was supposed to be rejected.

We set the target distribution as ($R \sim$ beta(3,9), $G \sim$ beta(9,1), $B \sim$ beta(10,10)).

2.2 Color selection task web application

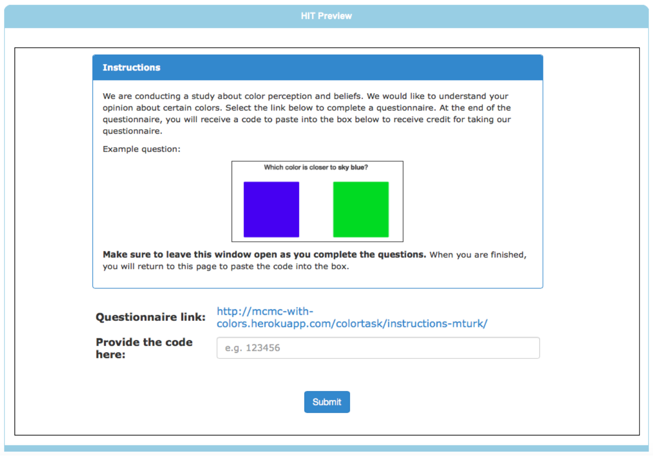
A web application built with the Django web framework enables subjects to give responses to the color selection task. The application is hosted at `http://mcmc-with-colors.herokuapp.com`.

Subjects are directed to a instructions page with a sample question before going to the main task, which consists of 120 questions. Subjects are presented with two colored squares and asked which color between the two presented is closest to a target color term. The target color term stays the same throughout the task. Standard HTML hex color codes corresponding to the RGB color space are used to render colors in the browser.

One square represents the proposal and the other represents the current MCMC state. After the subject chooses a color, a new proposal color is obtained (with the same proposal distribution as in the simulation), and displayed with the new current MCMC state. The position (left vs. right) of the colors are randomized.

The source code for the color task web application and simulations can be viewed at `https://github.com/kunanit/mcmc-with-colors`.

2.3 Obtaining data from Mechanical Turk workers



We are conducting a study about color perception and beliefs. We would like to understand your opinion about certain colors. Select the link below to complete a questionnaire. At the end of the questionnaire, you will receive a code to paste into the box below to receive credit for taking our questionnaire.

Example question:

Which color is closer to sky blue?

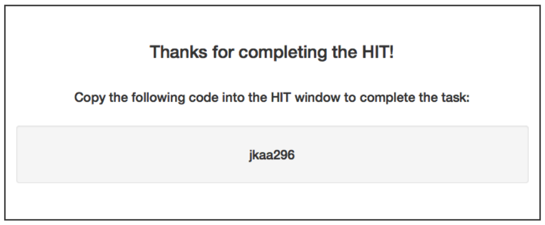
Make sure to leave this window open as you complete the questions. When you are finished, you will return to this page to paste the code into the box.

Questionnaire link: http://mcmc-with-colors.herokuapp.com/colortask/instructions-mturk/

Provide the code here: e.g. 123456

Figure 2: mTurk HIT redirecting to external web app hosting color task

A HIT (Human Intelligence Task) was created on Amazon Mechanical Turk (mTurk), a crowdsourcing platform, paying workers \$0.05 for completing the color task. Workers interested in the task were presented with a link to the color task web application. This way, mTurk subjects see the exact same task interface as non-mTurk subjects. After completing the questions, subjects were given a unique code to enter into the original HIT window (non-mTurk workers were not presented with a code), enabling identification of workers that completed the task. Data from 200 total mTurk workers was used for analysis, along with data from 8 non-mTurk workers.


Thanks for completing the HIT!

Copy the following code into the HIT window to complete the task:

jkaa296

Figure 3: After completing the questions, mTurk workers were given a code to enter in the HIT window.

Table 1: Number of subjects per target color

Target color	# subjects
blue	16
sky blue	51
crimson	40
purple	55
violet	46

3 Results

3.1 MCMC simulation

Table 2 compares the theoretical mean and standard deviation of the target distribution to the observed sample mean and standard deviation obtained from the 2 simulations (with and without simulated acceptance error). Separate histograms are created for the R, G, and B components of MCMC samples, overlayed with a kernel density estimate and the component's target distribution in Figures 4 and 5. MCMC without simulated error produced samples comparable to the target component distribution, even with the relatively short chain length of < 90 per MCMC instance. Introducing simulated acceptance error noticeably affects the MCMC sample distribution as shown in Figure 5. Acceptance error as the effect of increasing the standard error of the component distributions.

3.2 Color distributions of subjects

Representative MCMC sample paths from single subjects are shown in Figure 6.

For each target color, valid MCMC samples (samples collected after the 30th proposal) were pooled across subjects and used to estimate the color distribution (Figures 8-12), sample means and standard deviations (Table 3). Figures 8-12 show the pooled MCMC samples in a 3-D RGB space, along with sample distributions projected in the R, G, or B dimension. The color that corresponds to the sample mean of each estimated

Table 2: Comparison of mean μ and standard deviation σ between the target distribution and simulated MCMC samples

Distribution	μ_R	μ_G	μ_B	σ_R	σ_G	σ_B
Actual	64	230.4	128	30.7	23.2	28
MCMC samples	65.8	219.7	126.2	32.3	26.1	31.4
10% acceptance error	81.1	194	122.5	48.7	47	45.3

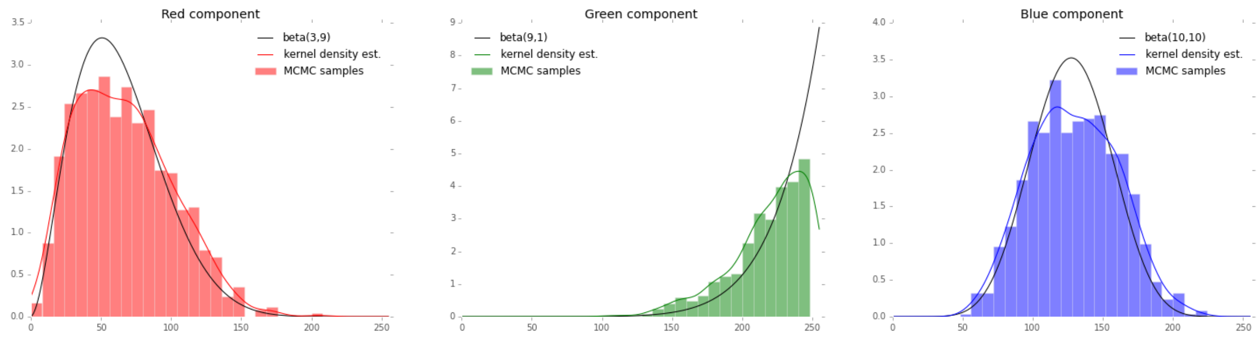


Figure 4: RGB component distributions of simulated MCMC samples, no simulated error.

distribution is shown in Figure 7.

As expected, the color distribution for ”blue” has mostly low values of R and G, and high values of B (Figure 8).

An interesting comparison to make might be to compare the distribution of ”sky blue” vs. ”blue” - Figure 9 actually suggests bimodality in the sky blue distribution, with a cluster corresponding to traditional ”blue” and a cluster corresponding to a lighter blue with a significant green component. On the 3D plot there is spreading towards the oranges, reds and purples (interestingly, the plot almost resembles a sky and sunset).

The difference between purple and violet is more subtle, but a slight hue change is noticeable when looking at the color means in Figure 7 (purple seems brighter than violet).

4 Conclusion

Using people as elements of a MCMC algorithm is an interesting combination of human computation and information elicitation, and provides a method for learning about from people’s subjective distributions on categories. We built a web application that presents color stimuli according to the MCMC algorithm, and used it in conjunction with mTurk

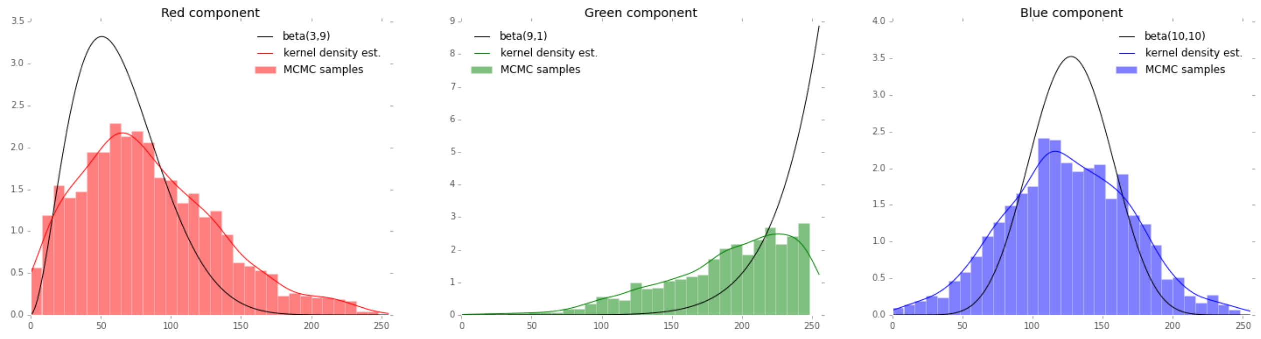


Figure 5: RGB component distributions of simulated MCMC samples, 10% acceptance error.

Table 3: Sample mean μ and standard deviation σ of MCMC samples from subjects, by target color.

Target color	μ_R	μ_G	μ_B	σ_R	σ_G	σ_B
blue	43.4	68.2	205.6	29	51.3	40.8
sky blue	95.3	133.2	198.5	64.1	68.7	57.5
crimson	162.9	90.9	89.4	64.8	72.6	65.7
purple	146.8	58.2	169	57	50.5	51.6
violet	133.1	74.5	157	67.1	58	60.7

to obtain crowdsourced data representing peoples' beliefs on specific color hues.

References

[1] Adam Sanborn, *Markov Chain Monte Carlo with People*, 2008.

[2] Adam Sanborn, *Uncovering Mental Representations with Markov Chain Monte Carlo*, 2010.

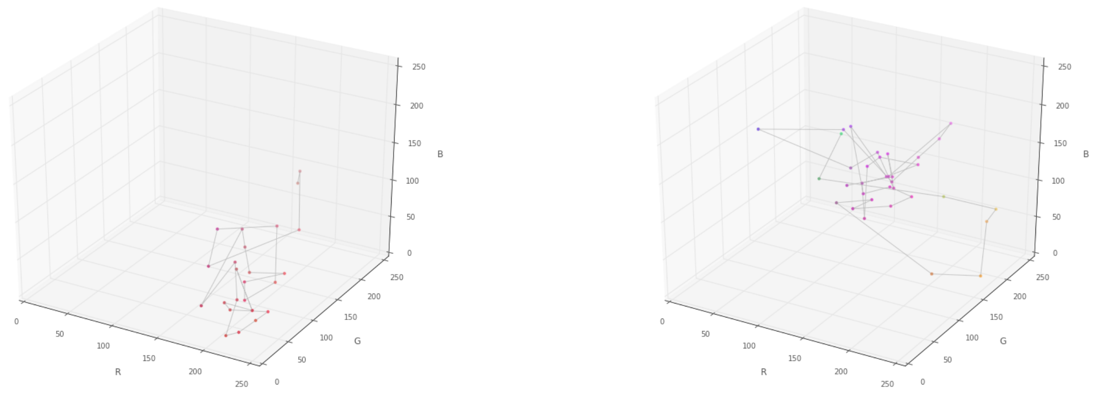


Figure 6: MCMC sample path produced by individual mTurk subjects. Target color: crimson (left) and purple (right).

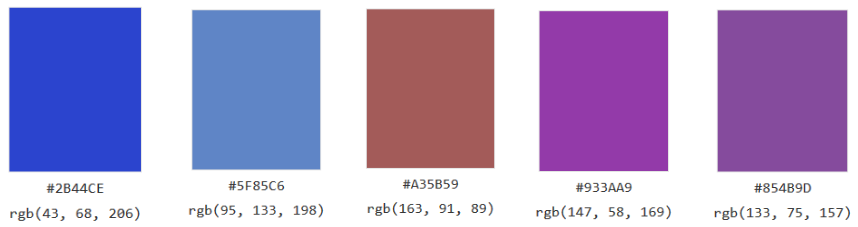


Figure 7: Average colors produced by subject MCMC samples. Target color (left to right): blue, sky blue, crimson, purple, violet.

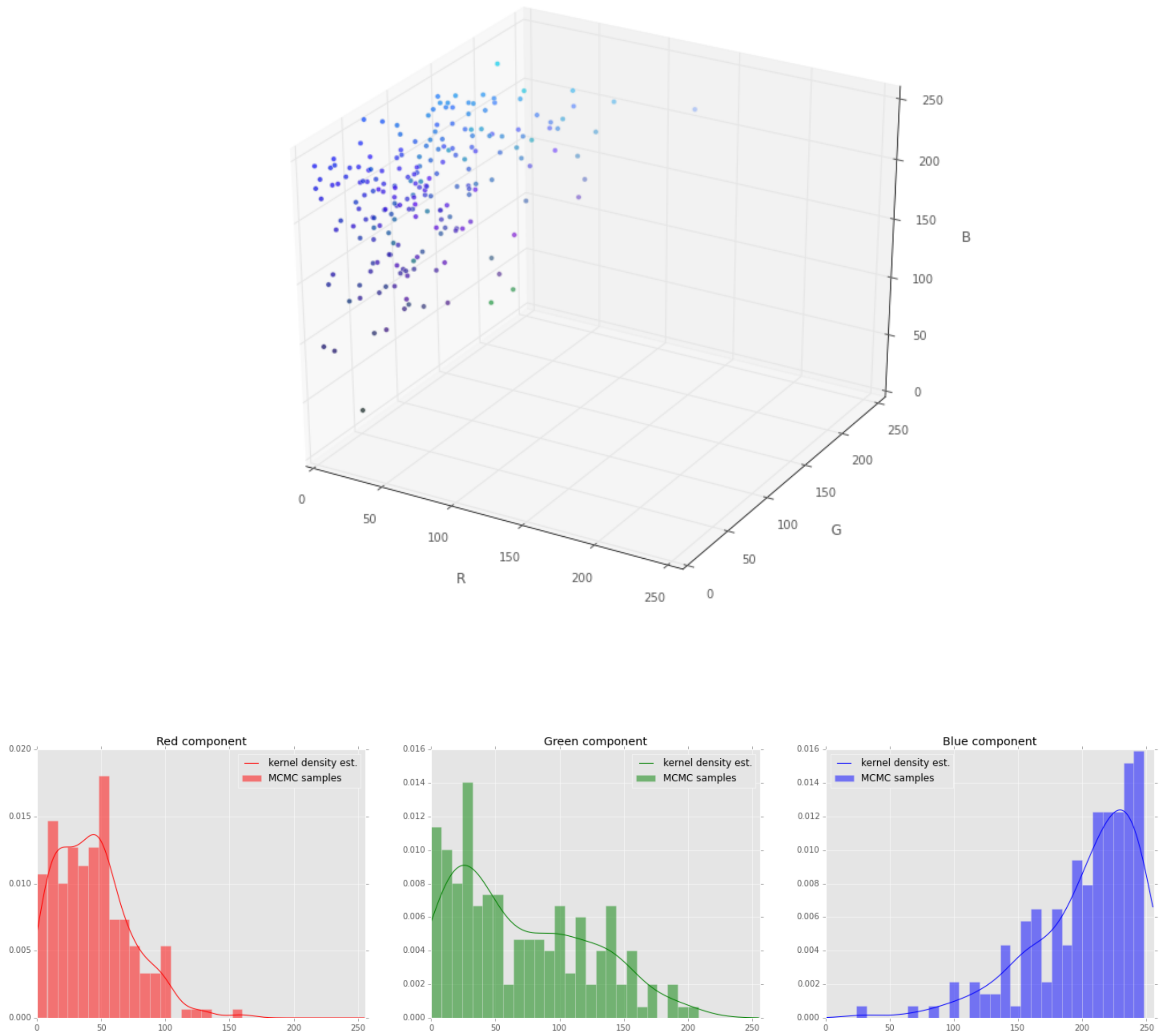
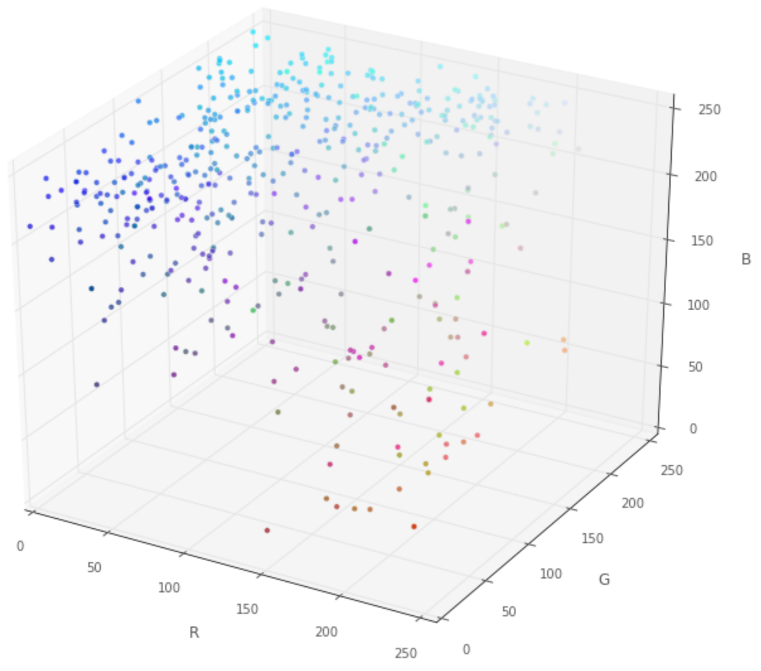


Figure 8: MCMC samples plotted in RGB space (top) and component distributions (bottom). Target color = blue.



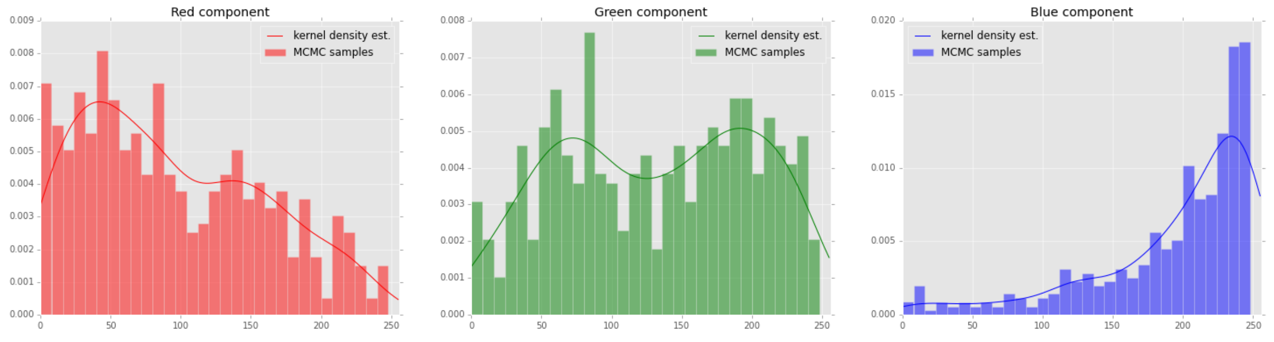
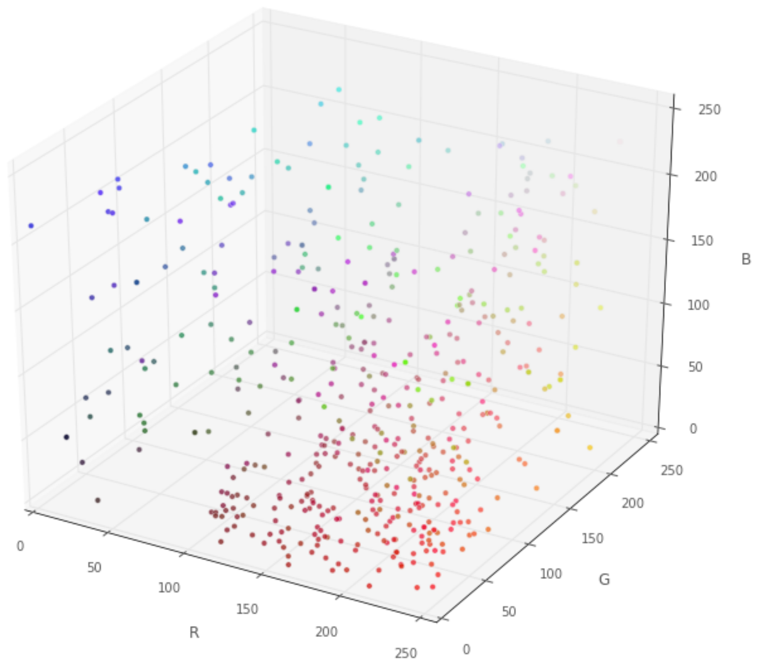


Figure 9: MCMC samples plotted in RGB space (top) and component distributions (bottom). Target color = sky blue.



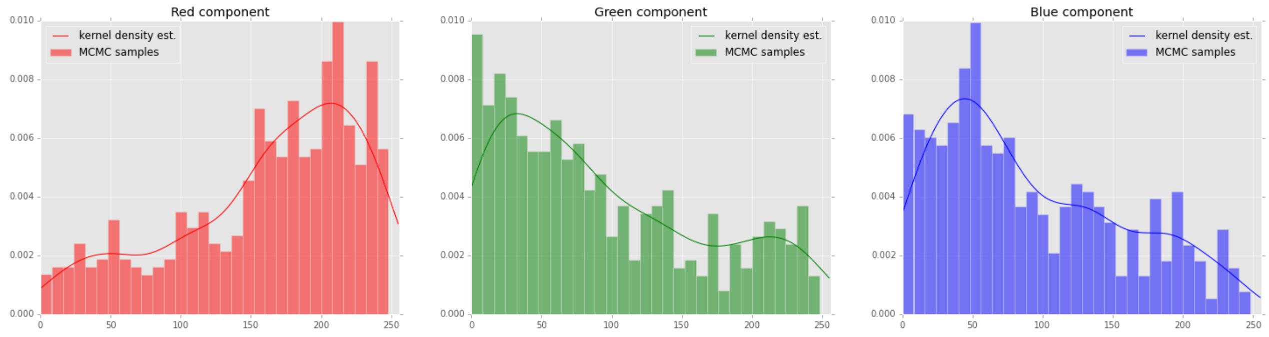


Figure 10: MCMC samples plotted in RGB space (top) and component distributions (bottom). Target color = crimson.

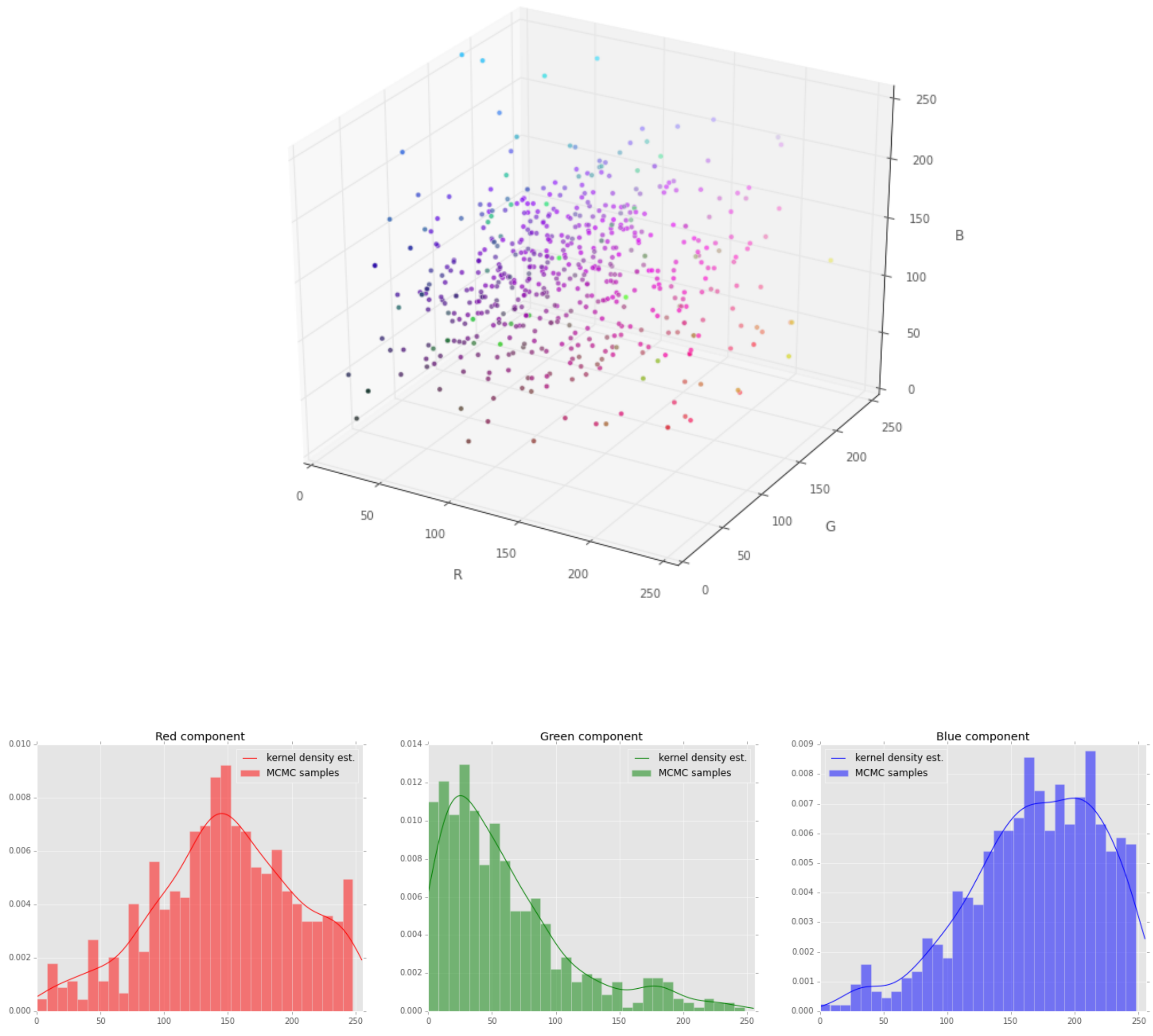


Figure 11: MCMC samples plotted in RGB space (top) and component distributions (bottom. Target color = purple.

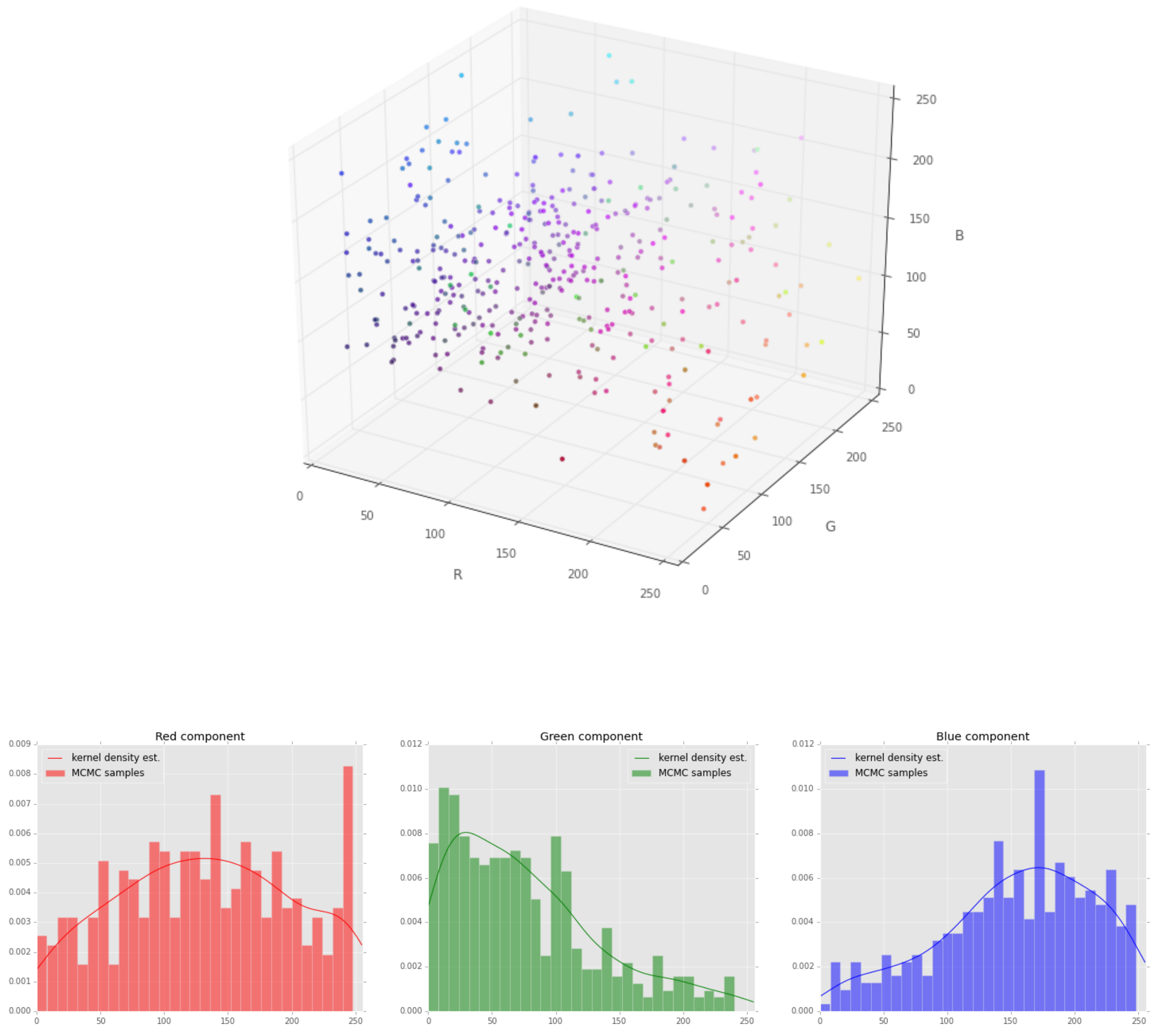


Figure 12: MCMC samples plotted in RGB space (top) and component distributions (bottom). Target color = violet.