The p value

This document is dedicated to explain the intuition behind the p value, which is an element of hypothesis testing. This will be done by simulation in MATLAB. Here, we will make use of the Student's t-distribution and fix the degrees of freedom to 20 and consider $20,000$ observations. The used script file can be found here on GitHub.

```matlab
%% Generate Student's t distributed values
rng(14)
n = 20000;
nu = 20;
nbins = -5:0.01:5;
generatedvalues = random('T', nu, 1, n);

figure;
histogram(generatedvalues, nbins, 'FaceColor', '#b3b3b3', 'EdgeAlpha', 0);
title("Simulated t distribution", "n = "+ n + " and \nu = "+ nu);
xlabel('t');
ylabel('Frequency');

% The code below is used to resize the plot
mp = get(0, 'MonitorPositions');
mwidth = mp(1, 3);
mheight = mp(1, 4);
scale = 0.45;

gwidth = scale*mwidth;
gheight = 0.75*gwidth;
x0 = 0.5*mwidth*(1 - scale);
y0 = (mheight - gheight - 84)*0.5;
fig = gcf;
fig.Position = [x0, y0, gwidth, gheight];
```

Running this section of code will give the following plot:

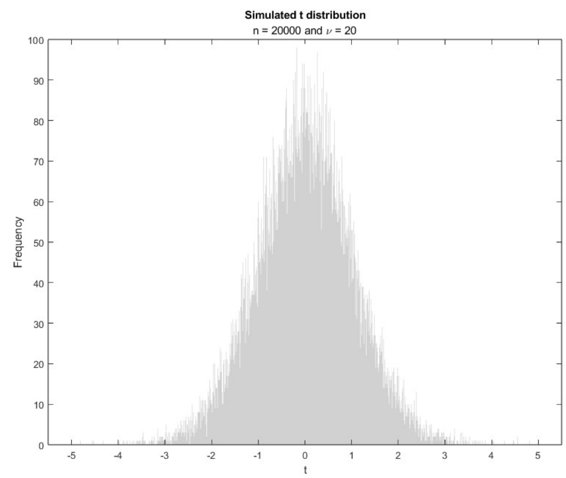

Figure 1: Frequency of 20,000 t-distributed samples

For this example, we will let the t test statistic have a value of 1.8;

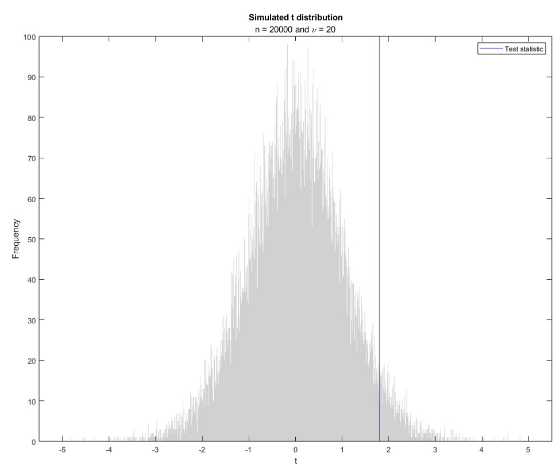

Figure 2: Frequency plot with the test statistic

The above plot was created by adding the following two lines of code.

```
tvalue = 1.8;
xline(tvalue, '-b');
```

The goal of hypothesis testing is to reject the null hypothesis, which is done by testing whether the statistic is extreme enough. For simplicity, we will consider a right tailed test here. The null hypothesis H_0 and alternative hypothesis are:

$$
\begin{aligned}
H_0 &: \hat{\beta}_1 \leq \beta_1^0 \\
H_1 &: \hat{\beta}_1 > \beta_1^0
\end{aligned} \tag{1}
$$

Since in the alternative hypothesis we have stated that $\hat{\beta}_1 > \beta_1^0$, we need to find all t-distributed values larger than the value of the test statistic (1.8). In the next code we therefore save all the frequencies of values larger than 1.8 and display them on top of the current plot.

```
%% The frequency of the values more extreme than our test statistic

% We first need to collect all the generated values that are more extreme
% (larger) than tvalue.

extreme = generatedvalues(generatedvalues > tvalue);

figure;
histogram(generatedvalues, nbins, 'FaceColor', '#b3b3b3', 'EdgeAlpha', 0);
hold on
histogram(extreme, nbins, 'FaceColor', 'blue', 'EdgeAlpha', 0);
xline(tvalue, '-b');
title("Simulated t distribution", "n = "+ n + " and \nu = "+ nu);
xlabel('t');
ylabel('Frequency');

legend('', 'Values more extreme than the test statistic', ...
    'Test statistic')

mp = get(0, 'MonitorPositions');
mwidth = mp(1, 3);
mheight = mp(1, 4);
scale = 0.45;

gwidth = scale*mwidth;
gheight = 0.75*gwidth;
x0 = 0.5*mwidth*(1 - scale);
y0 = (mheight - gheight - 84)*0.5;
fig = gcf;
fig.Position = [x0, y0, gwidth, gheight];

% The fraction of observations having a value more extreme than our test
% statistic. This is the simulated p value
manualp = length(extreme)/length(generatedvalues);

fprintf(['The fraction of observations that have a more extreme' ...
    ' value is equal to %.04f.\n'], manualp);
```

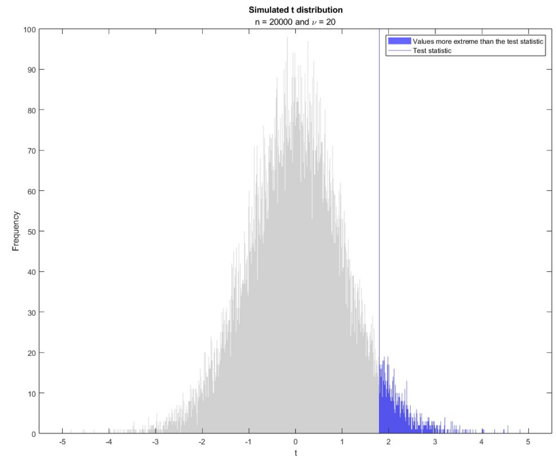

Figure 3: Frequency plot with all 'extreme values' shaded

The simulated p value is the fraction of observations having a more extreme value than the test statistic. In the current example the simulated p value is estimated to be 0.0435 or 4.35%, which therefore implies that about 4.35% of the total observations have a value greater than 1.8.

We will now extend this example to the case of the limiting distribution.

```
%% Limiting distribution
nu = 20;
tvalue = 1.8;
alpha = 0.05;

x = -5:0.1:5;
y = pdf('T', x, nu);

figure;
plot(x, y, '-black');
title("PDF of the t distribution", "\nu = "+ nu);
xlabel('t');
ylabel('Probability Density');
hold on

% Shade the density of all values more extreme than our statistic
pint = abs(tvalue):0.1:5;
py = pdf('T', pint, nu);
parea = area(pint, py);
parea.FaceColor = 'blue';
parea.EdgeColor = 'none';

% Display the value of our statistic
xline(tvalue, '-b');

legend('', 'P value', '', 'Test statistic')

mp = get(0, 'MonitorPositions');
```

```
mwidth = mp(1, 3);
mheight = mp(1, 4);
scale = 0.45;

gwidth = scale*mwidth;
gheight = 0.75*gwidth;
x0 = 0.5*mwidth*(1 - scale);
y0 = (mheight - gheight - 84)*0.5;
fig = gcf;
fig.Position = [x0, y0, gwidth, gheight];

% Calculate the p value from the cdf and compare this to the 'manually'
% calculated p value.

realp = 1 - cdf('T', tvalue, nu);

fprintf(['The p value is equal to %.04f,' ...
    ' and the manually computed p value is %.04f.\n'], realp, manualp)

% So, the p value is the total density of all values being more extreme
% than the test statistic.
```

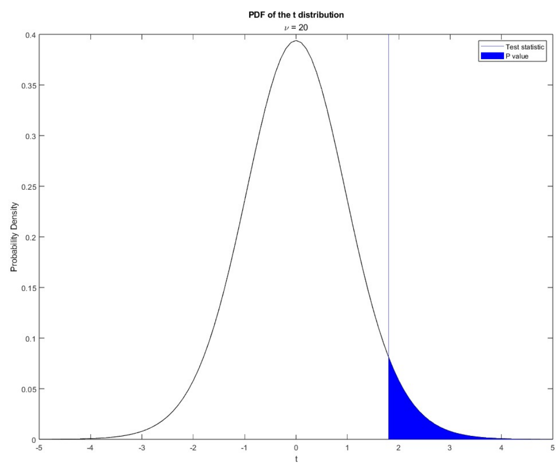

Figure 4: Limiting distribution with all ‘extreme values’ shaded

In this case, the p value can be derived from the distribution itself instead calculating the fraction like we did prior. Let $P(X \geq x)$ denote the probability that X is greater than or equal to x with $X \sim t_{20}$ (X follows a t-distribution with 20 degrees of freedom). Here, the p value is therefore equal to $P(X \geq 1.8)$.

The cumulative distribution gives the probability that a random variable X is smaller than or equal to some specified value x. This function is denoted by $F_X(x) = Pr(X \leq x)$.

Since $P(X \geq 1.8) = 1 - F_X(1.8)$ (one minus the cumulative distribution function of X evaluated at 1.8), the probability (p value) can be easily derived in MATLAB by using the built-in `cdf` function. In this example the p value is 0.0435, which is equal to the simulated p value.