Temporal skewing

huh??

Week 6a

I burned out an Arduino

- Arduino Nano’s 5V power supply is a wimp!! Don’t take current from it!!
  - Pulling ~50mA from the 5V pin burned out the Arduino’s voltage regulator leaving it partially shorted
  - The Arduino worked for a while longer, although it was stressing the USB port and causing other USB devices to frequently shut down
  - This was confirmed with a USB power monitor (e.g., <$4 https://www.ebay.com/itm/USB-Digital-Voltage-Charging-Tester-Power-Meter-Tester-Current-Charger-Monitor/174409336767)
    - Note: I bought a cheap one that was grossly inaccurate, my second one cost ~$25, has a gorgeous OLED display, and has been reliable

Bad connections …

- Trust nothing, verify everything … (and don't be paranoid …. Aaaaaaahhhhhh!!!!!)
- My black alligator clip wire had poor connections
  - Identified the problem measuring a resistor's resistance was unstable
  - Isolated the problem by simplifying the loop
    - Took out the resistor and 2^{nd} clip lead
  - Solved by soldering the wires to the alligator clips
  - Validated solution by wiggling and seeing no more instability on ohmmeter
- An earlier circuit revealed Vref was varying
  - Turned out that was not "the problem", but a *consequence* of drawing too much current from 5V pin, which eventually burned out the Arduino
  - I suspected it was a **proximal cause** because of experience with instabilities in other Arduino circuits
    - Thus I went down a wrong path …
    - My prior experiences apparently weren't as relevant as I thought

Good habits …

- How do you verify *everything*?
  - One at a time
  - Develop good habits, e.g.,
    - Every time your multimeter is set to ohms, 1st thing you do is measure the loop resistance of the leads and look for instability or offset
    - Try to predict measurements, extraordinary results require additional validation
  - Automate testing where possible

How fast can we vary the voltage pot?

Objective: Our Arduino program measures 3 voltages in rapid succession. If we adjust the power supply too quickly, then the first and last voltages will be different and our data will be skewed. Our goal now is to determine how fast is too fast to turn the knob.

Methods: *[Think about it before advancing to the next slide]*

How fast can we vary the voltage pot?

Objective: Our Arduino program measures 3 voltages in rapid succession. If we adjust the power supply too quickly, then the first and last voltages will be different and our data will be skewed. Our goal is to determine how fast is too fast to turn the knob.

Methods: Write an Arduino program that measures ~10 voltages as fast as it can, and prints them. We then turn the knob faster until we see voltage skew between the readings. The code is:

[try to write it yourself before advancing to the next slide]

Arduino Code

```
/*
  how fast can we turn the voltage pot
before we see voltage skew between the
first and Nth measurement?
  23sept20 BR

  hardware: LM317 output -> voltage
divider 10k/(10k+20k) -> A2
*/

#define VoutPin A2
#define VOLTS *(20+10)/10.0
#define NUMSAMPLES 10

volatile int Vout[NUMSAMPLES][3], i, j;

void setup()
{
  Serial.begin(9600);
}   // setup
```

```
void loop()
{
  // get data fast
  for (j = 0; j < 3; j++)
    for (i = 0; i < NUMSAMPLES; i++)
      Vout[i][j] =
          analogRead(VoutPin);

  // print the results
  for (j = 0; j < 3; j++)
  {
    Serial.print(j + 1);
    Serial.print(": ");
    for (i = 0; i < NUMSAMPLES; i++)
    {
      Serial.print(Vout[i][j]);
      Serial.print("\t");
    }
    Serial.print("\n");
  }
  delay(200);
}   // loop
```

Fun fact: This is the first time in many years that I’ve used 2-d arrays in C

File: howFastCanWeTurnThePot.ino

Results (turning the knob real fast)

1: 741	742	742	742	742	742	742	742	741	742
2: 742	741	742	741	742	742	741	742	742	742
3: 742	742	741	741	741	741	741	741	741	741
1: 421	421	420	420	419	418	417	417	416	416
2: 415	415	415	415	415	416	415	415	415	415
3: 415	416	416	415	414	414	414	414	415	414
1: 126	127	127	127	127	127	127	127	126	126
2: 127	126	126	125	126	126	126	125	124	125
3: 125	125	125	126	125	124	124	124	124	124
1: 206	206	206	206	206	206	206	207	207	206
2: 206	208	208	210	211	211	211	212	211	210
3: 210	210	209	208	208	209	208	208	209	209
1: 566	566	566	566	567	567	567	568	567	567
2: 567	567	567	567	567	567	566	566	567	567
3: 566	566	566	566	567	567	567	567	567	567
1: 741	741	741	743	742	743	742	741	742	741
2: 742	741	741	742	742	742	742	742	742	742
3: 741	741	741	741	742	742	743	742	742	742
1: 532	533	532	532	532	532	532	532	532	531
2: 531	530	531	530	530	530	530	529	529	529
3: 529	529	528	528	528	528	528	529	529	529
1: 168	168	168	169	169	170	170	169	169	169
2: 169	170	170	169	169	170	170	168	169	170
3: 168	169	168	168	168	169	168	168	168	169
1: 80	80	81	81	81	81	80	80	81	81
2: 82	80	80	80	80	80	80	80	80	80
3: 80	80	80	80	80	80	80	80	80	80
1: 264	266	266	266	266	266	266	267	267	268
2: 267	267	268	268	268	269	269	269	270	269
3: 270	270	269	270	269	269	268	268	267	268

Results (turning the knob real fast)

1:	741	742	742	742	742	742	742	742	741	742
2:	742	741	742	741	742	742	741	742	742	742
3:	742	742	741	741	741	741	741	741	741	741
1:	421	421	420	420	419	418	417	417	416	416
2:	415	415	415	415	415	416	415	415	415	415
3:	415	416	416	415	414	414	414	414	415	414
1:	126	127	127	127	127	127	127	127	126	126
2:	127	126	126	125	126	126	126	125	124	125
3:	125	125	125	126	125	124	124	124	124	124
1:	206	206	206	206	206	206	206	207	207	206
2:	206	208	208	210	211	211	211	212	211	210
3:	210	210	209	208	208	209	208	208	209	209
1:	566	566	566	566	567	567	567	568	567	567
2:	567	567	567	567	567	567	566	566	567	567
3:	566	566	566	566	567	567	567	567	567	567
1:	741	741	741	743	742	743	742	741	742	741
2:	742	741	741	742	742	742	742	742	742	742
3:	741	741	741	741	742	742	743	742	742	742
1:	532	533	532	532	532	532	532	532	532	531
2:	531	530	531	530	530	530	530	529	529	529
3:	529	529	528	528	528	528	528	529	529	529
1:	168	168	168	169	169	170	170	169	169	169
2:	169	170	170	169	169	170	170	168	169	170
3:	168	169	168	168	168	169	168	168	168	169
1:	80	80	81	81	81	81	80	80	81	81
2:	82	80	80	80	80	80	80	80	80	80
3:	80	80	80	80	80	80	80	80	80	80
1:	264	266	266	266	266	266	266	267	267	268
2:	267	267	268	268	268	269	269	269	270	269
3:	270	270	269	270	269	269	268	268	267	268

- Spinning the knob fast results in little change in 30 adjacent analogReads()
  - The biggest change was 421->414 = -7
- Serial.print() (at 9600 baud) and/or the delay(200) is slow, so the voltage changes a lot between each cycle through loop()
  - Biggest change was 532->168 = 364ADU * (4.8V/1023ADU)*3.2=5.5V

Conclusion: Its reasonable to expect there won't be voltage skew from the power supply when averaging 10 sequential reads on 3 channels.

Questions???

Don't Assume …

Measure!

Reflect

- I spent lots of walking, biking, shower time thinking of (easy) ways to test things like we just did
- There are many paths to the right answers
- Unfortunately, there are infinitely more *wrong* answers
- The more tools in your toolkit, the more choices you have
  - Here I implemented a rarely used tool, the 2-dimensional C array, to simplify indexing and keeping track of 30 data points
- Keep it simple
  - I stayed with ADU units
  - I didn't compute averages or statistics – why?
    - Because many statistics assume Gaussian or uniform distributions and the whole point is to validate the distribution of data
    - I can eyeball the stats
    - I don't want to hide outliers
    - I'm not going to do this test again and again … now that I know the answer

Reflect

- Activities like this pay additional dividends …
  - Not only did we verify how fast we can safely turn the voltage pot without skewing the data but we also …
  - Practiced programming
  - Got more familiar and intuitive with our Arduino
  - Added a tool to our toolkit …
- Teachers *hate* the question, “will this be on the final exam?” because learning something with that sole purpose is a waste of time and talent …
  - Take every opportunity to practice and learn new skills, add new tools to your intellectual toolkits, realize unconventional uses and applications of tools
  - Learn to innovate (create new connections between disparate things)
  - With the world changing so fast, we never know what tools might be useful for some novel task
- If there’s an *easy* way to test your understanding or theory, then *go for it*!! If you’re wrong, finding out sooner is much better than later!

Reflect

- Should we change `delay(200);` to another value??
- Any benefit to make it slower?
- Any benefit to make it faster?

- This is an example of "refactoring" – rethinking old code in light of new data …

Automating labs II.

- We didn’t discuss the human-robot component – that the LM317 pot has to be manually turned to vary the voltage while data is being collected
  - Forgetting to include such key steps or processes are an Achilles heel of automation projects
  - We will later build a circuit enabling Arduino to control the LM317 voltage
- Having the Arduino both control the apparatus and record responses adds huge automation capabilities

- But *first,* let’s partially automate report generation!!

Matlab's Live Scripts

- Live scripts let us mix text, graphics, data, executable code, and output, together in a single interactive environment called the Live Editor
- Make one in the editor by Editor → New → Live Script
- Toggle between Text and Code mode with option-return
- Pictures can be pasted into Text blocks
- Execute the code with cmd-return
- Icons on the top right scroll bar enable displaying of results below text or on the side, and hiding the code
- Save as → Export to (Word, HTML, PDF, Latex) generates corresponding files for sharing

Example:

Measuring Hfe of a transistor

18 Sept 20 B.R.

Objective

This is our first transistor experiment, and our goal is to measure its characteristics and current gain over a range of input voltages and loads.

Methods

We built the circuit in Fig. 1, the transistor is a 2N3904. The two resistors on the left are voltage dividers ensuring the Arduino measures <5V for LM317 outputs up to 11V. Rb, the base resistor, turns the LM317 variable voltage into a variable base current, Ib. 3 values of RL were tested.

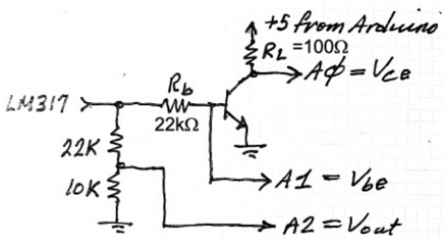

Figure 1. Schematic of the test circuit and connections to Arduino's analog inputs A0-A2.

Arduino Code

This C code was copied from our first I-V curve experiment and expanded upon. It's commented below (with block comments, %{ ... %} so Matlab displays it as code but doesn't try to execute it. In the I-V source code, we did as little as possible on the Arduino, and returned data in ADU (analog to digital units). More confident programming in C now, we put more functionality in the Arduino in order to have to do less conversions and data processing in Matlab.

```
%{
/*
  use PWM to vary transistor base current
  measure base and collector voltages and currents
  NOTE: make sure LM317 is no more than 5V
  17sept20 BR
*/

#define VcPin A0
#define VbPin A1
#define VoutPin A2
#define VREF 5.3 /* volts on Vref pin */
```

...

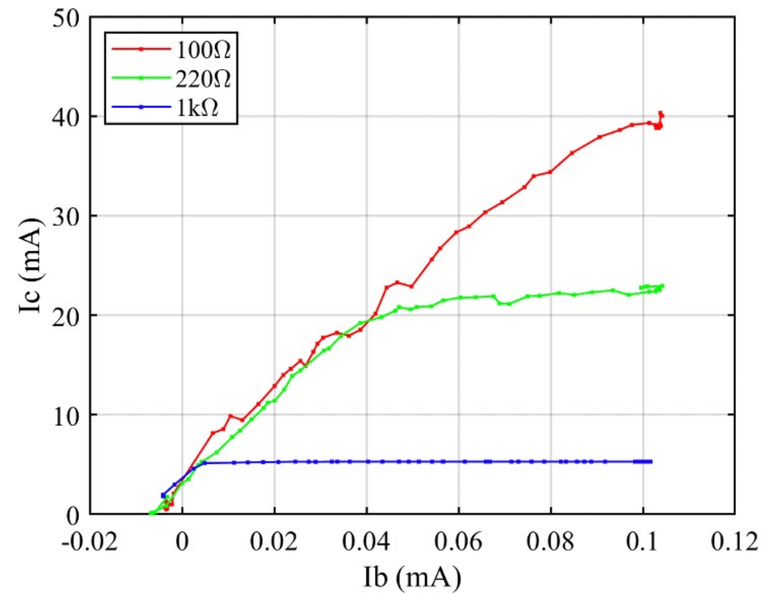

Figure 4. Collector vs. base current for 3 values of collector resistor show similar initial behavior.

Discussion

The transistor exhibits 2 regimes in the figure above: one where Ic is proportional to Ib. We call the constant of proportionality hfe. The other regime is where Ic is constant, and we call that regime "saturation", i.e., the transisitor is saturated meaning further increasing Ib does not increase Ic. That's because Ic is limited by the external power supply and the collector resistor: Ic = (Vcc-Vce)/Rc ~ (Vref-0.3V)/Rc.

```
Rc = [100 220 1000];
icSaturation = (vref-0.3) ./Rc * 1000 % mA
```

```
icSaturation = 1×3
   50.0000   22.7273    5.0000
```

Pretty good agreement with the curves.

What about the slope, hfe? Matlab's polyfit is a great tool to find the slope ... but we don't want to use it blindly, just on the data before the saturation "knee".

...

Update the data and the calculations and figures automagically change ...

(BTW, noisy red & green curves from Vref instability, from too small R_L)

Alternatives

- Jupyter (www.jupyter.org) and Processing (www.processing.org) are two similar environments that merge programming with reporting … check them out!
- The typewriter was replaced by the word processor (good luck finding one now)
- Will these integrated environments replace the word processor for some scientific writing?
- Consider that a word processor isn't ideal for scientific publications