The hypatia project

Mauri Favaron – 4th July 2018

# The problem

Data, either big or not, are an important basis for any decision process.

The more this is true, the more we can also say data are precious, and their quality important.

Data value (and quality) is especially high in fields like “making science”, engineering design, policymaking, and all fields where decisions may have a profound, or pervasive (or both, or more) impact on everyone life.

In this note I will limit to the process of “making science”, in part because I’m a bit more familiar with it, and in part because some of the general concerns related to data quality get an especially intuitive “fabric” when declined in science-making terms. You, as a sensitive reader, may extrapolate to your own life, goals and experience.

To make things more practical, and to get an easier understanding of the things at stake, my feeling is to focus on one aspect of “making science”, of the many existing. This may not be fair to scientists (who, as any other person in the world, are quite afar from the implied single-mindedness in selecting just one aspect), but on the other side helps, by allowing concentrating on some essential aspects and leave all unimportant out, and by setting a personal frame.

So, let’s imagine a bit. The science-making task in which we’re involved is, say, performing experimental work aimed at testing the effectiveness of air pollution mitigation by the use of a new kind of adsorbing barrier, to be mounted along highways.

The research, and its results, will be published in one or more papers on some important scientific journal. So, you can expect in advance our work will be closely scrutinized by some journal editor and peer reviewers, who may reject our paper altogether, or ask for changes (corrections, clarifications).

Let’s assume a large part of the experimental work will be made by collecting field data, and then comparing them with help of some defined statistical procedure. These field data span quite a wide angle, from real-time recordings of some pollutants concentration (let’s say, NOx, CO, C6H6), to end-campaign lab analyses of bio-monitoring specimen, to meteorological data like temperature, relative humidity, mean wind vector, air turbulence, precipitation, inbound short-wave radiation, and, many more.

Most of these data (let’s say, all but bio-monitoring and passive sampler lab analysis figures) are collected automatically by a number of “data loggers”. These devices are sort-of field computers, collecting many “elementary” measurements and hourly-averaging them.

Our paper(s) will then be, in the very end, reports on the hourly (and end-campaign) measurements we got from field, plus their interpretation with the formulation of conclusions about whether data support the fact our new barriers is effective against traffic-bound pollution, or not.

As far as I’ve seen, most often the reviewers will focus on methodology (that is, where data have been collected; with which sensors; for how long and how frequently; which statistical tests have been used, and whether they have been applied appropriately).

One thing will reviewer *not* question (unless the paper is so awkwardly written to make even them suspect): the (hourly-averaged, or end-campaign) data themselves. The instinctual assumption is that if good reputation sensors have been used appropriately, then, automatically, the numbers will be trustworthy.

(I’ll not mention here one very important point, that is, error bands. Measurements, in the real world, are subject to some degree of uncertainty, and in any serious paper the uncertainty figures are always given. For now, let’s imagine we live in so an ideal world that the standard deviation of measurement replicas is always zero.)

But here, a big problem lurks: data loggers themselves!

Data loggers are electronic pieces-of-art. They are typically highly robust and, that’s very important psychologically, also *look* so. They must resist extreme field conditions, and so this evident robustness is not unexpected. Quite on the contrary.

In addition, some of them are really beautifully made. A model I often use (which imposed itself as a sort of de-facto standard in meteorological measurements) is made of two interconnected units: a sturdy “wiring panel” to which you connect sensors, power supply and transmission devices; and, a curvy stainless steel “core” which looks so hard it could sustain the shock of a direct lightning. Believe it or not, the data logger manufacturer worked carefully not only on the shape, but on the *color* of the core (an early version was steely-gold), and its fabric. Any detail is designed to improve robustness *and* visual and tactile aspect, in confirmatory manner.

Such a rugged, wonderfully functional, minimalistic sculpture never and never could provide “wrong” data, isn’t it? And this is what many reviewers (we included…) tend to feel.

Unfortunately, things are not that simple when data are collected automatically, and aggregated to form time “averages” (and standard deviations, and …). This are some reasons, from my own experience (other people may have encountered others more):

* Electrical signals entering the data logger may not reflect accurately the “measurements” coming from sensors (and, these “measurements” may not reflect reality so that accurately – but this is another story we’ll not deal with). I’ve met two main causes:
  + Transducing a physical reading from a primary sensor (say, a cup anemometer instant revolution rate) to an electrical signal involves errors, delays, selective attenuations, and a really vast bestiary of other co-causes of “uncertainty”.
  + And additionally, once produced, an electrical signal may mix with noise (because of, say, electro-magnetic fields nearby, but also because of imperfect connectors and wires, unbalanced impedance, and more more more).
* Once in the data logger, electrical signals go digital through a Digital-to-Analog converter (in sophisticate data loggers this step may be preceded by an amplifier and an analog filter). This conversion is affected by various “errors”:
  + Finite resolution (the digital word holding conversion results is an integer number constrained to DA specific limits; diffused DA chips and circuits have output word size of typically 8, 10, 12, 14, 16 or 24 bits (the smaller, the faster sampling rate).
  + One or more of the least significant bits may be “uncertain” – that is, presented with the same voltage, a DA circuit may yield different results on different times. Said differently, some degree of non-determinism is present in digitalized data.
* The data logger, as beautiful looking and polished as it looks, nevertheless is a mineral object. As any other rock, it weathers as soon as it comes in contact with the Earth atmosphere. Weathering will eventually result in a general system failure, maybe after decades of “regular” work. But before of that, it will manifest with discrete components (e.g. resistors in signal amplification stage) changing their values respect to their starting nominal state. This is a strong argument in favor of data logger calibration, in addition to sensor calibration; in practice, data logger calibration is performed very seldom, if any at all.
* When collecting an “elementary” data item, a data logger in reality might perform many DA conversions, include the plausible ones in a short-term statistic (think a mean, for example), and yield the latter as an “elementary” reading. Details of plausibility checking and data censoring policy are usually un-disclosed, and their (sure) existence translates into an additional quota of uncertainty.
* Another (similar but not identical) data processing issue is, elementary (aka “raw”) data are also censored and discarded from “hourly averages” if failing to meet some quality criteria. Differently from elementary data, in this case the details of censoring process are usually known publicly.
* After elementary data have been “read”, they are most often converted from “counts” (integer values) to “engineering units” (floating point). It is these floating point values which are aggregated in averages and other statistics. Conversion to engineering units may have two important issues:
  + To begin with, a precision loss may occur. This is especially evident if the floating point word length (and internal structure) is smaller than 32 bits, as happens with some data loggers to save memory space. (In most cases I’ve seen this is just a minor problems; yet, fitting a non-scaled pressure in hPa units to a non-standard 16 bits floating point word proved to be impossible.
  + Second, the response equation describing the correspondence between count and engineering unit values may be complicate, or contain uncertainty elements, or difficult to translate in the data logger’s conversion abilities. In most cases, conversion is performed through a straight line, characterized by a multiplier and offset. But multiplier and offset are themselves finite-precision floating point numbers, and so even if the linear correspondence is perfect, its actual implementation may be not.

All these things, and many more, occur. Simultaneously.

On these days, we face a potential issue more: many components (namely, sensors) which once were brutally simple (like for example a resistive thermometer) now more and more are replaced by objects containing their own tiny micro-controller. We do not any more let the data logger “talk” to the sensors directly, but rather interchange data with a dedicated computer. With these smart sensors, as they’re often named, our margin of control reduces somewhat, and the problems I said of above may actually magnify (although hardware design tends to get simpler).

That assumed, and with little hope of being contradicted, we can choose many lines of action. One (indeed popular) is to basically ignore the problem. If something strange occurs, that’s a fault in the data logger manufacturer quality assurance, and their responsibility. Such a head-in-the-sand approach might not work well if data were really critical.

Or, one may assume the opposite approach, of constructing a special purpose data logger tailored to the research needs, in which all steps of the “averages” formation are perfectly transparent. This (bit idealistic) approach demands however an unlimited budget, along with the need to hire people carrying skills outside the typical range of research team member competencies (when it happens, expect major communication problems).

Still another approach – the one I’m suggesting here – is to improve current data logger technology allowing it to produce not only “averages” (or “raw”) data, but, also, a set of quantities allowing to understand how the averages have been formed. These additional diagnostic data may not be provided always (as this would make data sets cumbersome), but should in my feeling available on request. Data loggers may then be produced professionally, by dedicated companies, at a much lower cost and high reliability.

Whatever we decide to do, it would be better we maintain our awareness firm on a fact: the data loggers we’re light-heartedly using in our daily research work do impact on the quality of results. We take a great care in ensuring the lab technicians clean all test tubes and glasses as carefully as possible, so, let’s consider the many other “lab technicians” we’ve never seen and knew, who decided the details of our data logger innards.

# The Hypatia Project

## Main objective

And so, here we are!

Once established a bit of context, it is time to state what we desire.

In my case, the Christmas gift I’d really love receiving from the datalogger manufacturers is the notice of an affordable data logger able to produce “transparent” data, that is, data whose formation process is know enough to allow me, my colleagues and our reviewers to check the figures we’re to publish in the next fundamental paper bear an objectively resemblance to sensor outputs.

Making this desire into a reality might realistically not come from data logger manufacturers, who have to sell existing massive stocks of “non-transparent” loggers, nor come from scientists alone, who know about data acquisition engineering informally. Co-operation is essential, along with the (imagine inevitable but healthy) confrontations which could arise from distant points of view.

The project main (and final) objective is then to produce a *specification* of a data logger dedicated to scientific uses. Ideally, this specification, made by good will people and organizations, might evolve in a standard proposal.

As a minimum, a measurement of success of the Hypatia specification (or standard) would be that at least one (possibly some) data logger manufacturers agree in supporting it, by realizing Hypatia-compatible devices. Would that happen, then I’m afraid we would need a certification and labeling process… ☺

## Secondary (but not less important) objective

A specification (maybe a standard) is surely a cozy, reassuring and immediately useful tool.

But in itself has little power to change things, if people is not aware of the problem it tries to solve, and the perils of not solving it.

So, we need to communicate. To build awareness, and diffused knowledge.

Beginning with the ones who by their young age and still-unconstrained mind, can more easily grasp the novelty and, more important, implement it in a reasonable time (I will not declare my age here, but, stay assured, my times of flexible mind have long gone; as a glimpse of my scale of problem, consider I began writing code on a Honeywell Level 6 with the executive program (today we’d say “The Operating System”, but on those times this word was a bit inappropriate); before even, I spent one year being the “computer” myself – my Arithmetical-Logical Unit being an all-but-programmable hand-held calculator: if you want, I may still show you the “key-to-press-sequence” big notebook I filled for my own use).

Technology, to date, helps. Some of the varieties of data “average diagnostics” coming to my mind demand quite a large amount of floating point computations (am actually thinking to bootstrap algorithms…), and just a few months ago it would only have been a dream to implement them in real-time, at a sustainable energy cost. But now the Cortex M4F family is mass available, with development systems costing around 26$ (say 50€ after shipping and taxes). This opens up immense possibilities, totally unthinkable during my early (Jurassic!?) times.

In my experience, the field of data logger engineering tends to attract people who depart quite a bit from the image of the typical nerd. Might say, somewhat less ego-centered, and a bit nicer. I feel, for good reasons: first, data loggers are embedded systems, whose measure of success is to operate under nominal conditions without anyone noticing them. So, the more well your logger will run, the least other people will consider it – and, by transitive property, you, the engineer.

Another point I feel conductive is, an engineer crafting a data logger does *not* know in advance what application the data it produces will have. This is a very big responsibility, demanding a people-and-technology mindset. Thinking in advance, for example, to who will program the logger for some specific applications, will install or maintain it (maybe in Antarctica, with gloves you can’t remove – it happened to four of my systems, and will likely happen to you, maybe somewhere terrible). And, finally, decide whether using your numbers or not.

A good place, then, to deal with engineering, people, and engineering-for-people.

So, the Hypatia project shall have a very strong didactical inclination.

And this inclination will precede in time the final desire, the Spec.

## “Hypatia”: why?

Maybe a “who” would be a better starting point, would not she be so well known in our times.

I mean, Hypatia of Alexandria, the philosopher, astronomer and mathematician who lived in Egypt on the fall of the Roman world (around 370 – 415 AC).

Her fascinating figure, and tragic death in the hands of religious zealots, are well known, and there is no need I add my considerations about.

I would like to limit to only one point, in my view highly significant: the renown (and feared, by the powerful) Hypatia’s public philosophy lessons. From time to time – weekly at worst – Hypatia and her disciples went to the streets of Alexandria, and debated of philosophical themes together with all people who wished to attend, whatever their condition.

This surely was a revolutionary act, in an era when hoarding knowledge and releasing bits of it to common people was a primary source of power, by the government, and by the then growing church. Many biographers trace to Hypatia’s proclivity to share (she, a *woman*), instead of control, the “secrets” of nature and reality one of the causes of her execution.

And, also (this is the face I love more) deeply, warmingly humane. The very idea that knowledge is constructed (with hard work and, we’d say to date, “huge investments”) to be diffused, freely, without obligation, just because it elevates our spirit, and build opportunities. Maybe, not given to everyone: just to whom loves receiving it. Without constriction.

This beautiful idea deeply resonates with our modern mind-set, but in Hypatia’s time it really stood up. So, as little as we can be, and as small our desire of transparent data might seem, Hypatia is one of our ideal ancestors. One of the very first Modern Minds, and one of the first to affirm, with bravery and determination, what we also believe – and take for granted.

# First glimpses

## “Transparent data”

A Hypatia project compliant data logger is devoted to producing “transparent” data, or so I’ve said.

As the project is still in its very beginning phase, maybe some clarification is necessary –surely I need it.

But, after some time (not that long – about one year) of reflection an intuitive idea presented herself to me. I try giving her some communicable form.

* Transparent data (data sets) may exist in two (more?) forms:
  + Raw data
  + Averaged data.
* A third level of data (closest to the analog-to-digital converter) is supposed to exist: individual readings; they are not required to be transparent.
* Transparent averaged data are built from transparent raw data by means of an entirely specified algorithm, whose realization(s) is (are) publicly available in the form of open source code. I intend “averages” in quite a broad sense, close or overlapping to the meaning of “statistic” in techy descriptions:
  + Sample mean
  + Sample standard deviation
  + Sample skewness and kurtosis
  + Sample quantiles (any of the 9 definitions in common use – I’m thinking to the nine modes in the R language help about quantiles)
  + Centered moving average
  + FFT
  + …
* Transparent raw data are built from (non-transparent) individual readings using specified algorithms, available in open source. Among them, for example:
  + Centered moving average.
  + First order time domain filter
  + Higher order time domain filters?
  + FFT based smoothing?
  + Discrete wavelet smoothing?
* Transparent data (both raw and averaged) present their nominal value, *and* accompanying values allowing to understand (possibly, validate) measurements. My ideas are not yet clear on which could be a “minimum” set of those quantities, and whether this set is the same for all the statistics, or not. Things of great usefulness might include:
  + Estimation of noise, measurement uncertainty, …
  + Sample of tributaries? Like e.g. raw data for averaged, and individual readings for raw data?
  + Confidence limits based on data-driven empirical distribution (bootstrap method needed?)

As you can see, there are plenty of things to decide, and more even to review.

## The pilot implementation

Developing a specification in abstract can possibly be made, but, in my opinion, it could be difficult to evaluate its design and find unanticipated problems.

This is the main reason I imagine the Hypatia project accompanied by an ongoing development of a pilot, or set of pilot implementations.

Given the nature of the technical elements of the Hypatia project, I felt quite confident an Arduino derivative could have been enough. After some preliminary check, which led me to exclude the “small” 8 bit AVR microcontroller, I finally found two systems which look viable hardware platforms: the “little” SAMD21 based AdaLogger Feather32, and the much larger, SAMD51 based Metro M4, both produced by AdaFruit Industries, and widely accessible. Other actual platforms may also be considered however; my suggestion, based on code I’ve already written, is to stick with Cortex M0 or larger processors, and to avoid 8 bit variants: calculations needed for a fully Hypatia-compliant implementation may become quite extensive…

The choice of these 32 bit architectures is not coincidental: the SAMD21 and SAMD51 are currently used in data logger industry as sort of reference processors. Students cooperating to the pilot development would then have an opportunity to gain experience on what’s currently used in the “real world”.

For the pilot(s) to be entirely effective, it would be nice if the pilot functionalities are incorporated in source libraries, allowing re-use and refinement.

Pilots may be developed within labs, coding weeks, and the like. Having multiple pilots implementing the same set of requirements, giving freedom to test various implementations, and comparing them, could be better made in a cooperative spirit than competitively (at least, that’s how *I* would feel better). Requirement implementation might be conduced as a sort of role play, in which students are lead to explore the ramifications of crafting complex code so that other team members and prospective users can really benefit.

## What exists already (on 4th July 2018)

To date, some Arduino sketches are already available, with accompanying documentation, and reference to a case in which temperature and relative humidity are gathered through a DHT-22 sensor, using an AdaLogger Feather M0.

The sketches are numbered, and attempt to solve an increasing number of problems related to data logger engineering, like:

* Gathering single data from the DHT-22 using (through a library) its bit-bang protocol.
* Time-stamping gathered data, allowing to position them in time (with the user interface complications arising from the need of initializing the system’s real time clock with sensible date-time values).
* Writing data to an SD card, used as a sort of easy-to-use mass memory.

Consideration of various things like operating modes, and their selection through a simple switch-based interface, are also made.

The largest sketch has the capability to log time stamped temperature and relative humidity data to a fixed name file using the imprecise timing allowed by standard Arduino library calls. It implements a functional, although minimal, and surely not Hypatia-compliant, logger, and may be used as starting point for lab activity. You may find them in the GitHub project space: feel free using them.

I’m not entirely sure I’ll stop at this implementation level, however. I’m considering adding two other important (although not as essential as data logging itself, in this moment), like

* Simple but “real” real-time scheduling of data acquisition.
* Writing data to multiple files, whose names contain information about date and time.

Neither of those functionalities would turn the initial logger into something Hypatia-compliant, but could solve common data logging related problems which cannot be avoided.

Anyway, it is now possible to start concentrating on data quality assessment and improvement, thus entering the real full realm of Hypatia.

Anyone willing to help and contribute, at this point, is welcome.