Analog Computers:
Rise, Decay and Afterlife

by Nikita Timokhin

Table of Contents

Intro

As of current year 2019, computers have reached into every sphere of human life. From a scary, clunky machine, only a few of which exist in the world and accessible only to privileged scientists and military generals, through a bleeping box of mystery of which only computing enthusiasts dream and know, to small, handy devices responsible for both texting our friends and handling aviation and spacecrafts that we know today. Scientific uses of computers nowadays barely have a part in overall computer usage, and mostly they are, of course, used by "user" people of all sorts: we need games, we need social networks, we need advanced text editors, and so on, and so forth – and when we need it, we need it right now. That's why all modern home computers are easily able to multitask and are able to store and execute different programs at a click of the mouse. But what was it like way before win95, DOS, and other advanced systems? And even before (almost) all the computers were *digital*?

This is where we encounter an extremely interesting world of analog computers. Once used by scientists, then by students, and nowadays almost abandoned and forgotten, these machines have a whole lot of history to dig into, and theoretically have (and deserve) a chance for afterlife. In this essay, I am going to cover three main topics: history of analog computers – the working principle and a bit of dry theory, the history - rise, "golden age" and death of analog computing, and the chances and profits of it in modern age of digital supercomputers.

A bit about the technical side

"'Analog' derives from the Greek word 'analogon' which means 'model'. And that's exactly what an analog computer is: A model for a certain problem that can then be used to solve that very problem by means of simulating it."
-from "**Why Algorithms Suck and Analog Computers Are The Future**" by Brend Ulmann

So far, I have used the term "analog computing" tens of times, but most people are not familiar with this type of computers and the way they work – so I decided to give a bit of information on electric analog computing.

In digital computers, we have the CPU, the RAM, all kinds of interfacing chips and periphery, and tons of input-output devices. In analog? Forget about all of that, and rembmer – it's a computer because it computes a result.

Analog computers were used for scientific purposes at their better age, so they are meant to solve problems. A problem has initial conditions, the description of the case, and the results. Being the next step from mechanical computers, electric analog computers operate with continuous electrical signals: 'numbers' are represented as parameters of the electric current and are operated with by analog circuitry, mostly – discrete parts, such as: resistors, capacitors, inductors, mechanical switches and keys, in some cases – transistors, diodes and operation amplifiers, which form *modules*, each specifically designed for one single task: summator, multiplier, negator, inverter, divider, differentiator and so on. Below, I am going to briefly explain how this was done on 'classic' analog computers.

The task – the process, the result of which needs to be known, is modeled by interconnecting the modules in a specific, suitable way by plugging *patch cables* into corresponding *patch points* – sockets (usually banana plugs, but can be any type – even 3.5mm audio jacks we all know) that are placed on the *patch panel.* On figure 1, you can see such a patch panel. The task of creating a model out of these tiny modules is, indeed, time-consuming and requires special knowledge; yet there were some common guidelines and techniques that made the process faster.

The initial conditions – since they are values of some variables – are 'numbers', hence are usually represented as voltages. A special (and quite common) set-up of a part called potentiometer as a voltage divider allows the user to dial in the voltage they need simply by turning a knob.

The results are obtained on a device that measures a needed parameter: it usually is a voltmeter or an amperemeter, but may be something else. The more precise the device is, the more pinpointing results one will get from an analog computer. They can also be used to monitor the initial conditions before launching the system.

Fig. 1: Heathkit Analog Computer H1, an education-purpose machine built in 1956

Looks unnecessarily hard, tedious and complex so far? That's because it is. But at those times, such computers were the first choice for hard tasks: no other machine could solve differential equations as quickly and as precisely as these beasts.

“Unlike digital ones, analog computers don't count but measure (input) parameters. Thus an anlog computer does not even recognize the complexity of a differential equation – it solves even non-linear ones where there does not exist any approach very quickly by simply drawing the graph of the solution. On the other hand, analog computers are not suitable for solving literal equations and suitable for commercial calculations not at all.”
-from “Analog and Hybrid Computers” by Technikum29 museum

Omitting the description of ways to model problems, I would like to focus on the “analog computers don't count but measure (input) parameters” thing. I advise you to re-read it and ask yourself if you understood it right. When I first tried taking on what analog computers do and how are they potentially more precise than digital, it took me a deep dive into a ton of articles to get it. But, since you read this paper this far, I will explain it right there.

Modern digital computers store values in form of bits. Imagine a situation – as weird as it sounds – when a maximum number of bits a number can occupy in a computer is 3. You can have 000b, which is 0 in decimal, 001b – 1d, and so on until 111b – 7d. But what if you want to store an 8? Of course, you add one more bit and get 1000b. This way a byte, for example, can handle just 256 values – from 0 to 255.

Now, let’s shift this situation to floating point numbers. Imagine the maximum number you can store – say, 255 (11111111b) for a byte-long integer, is our 1, and 0 (00000000b) is our 0. All the values inbetween are *fractures*. Since we have just 256 different values to use, we can only store 254 quantized values between zero and one, with a step of 1/256 (approx. 0.0039). So we can have a value of 0.0039, and then – one step up – 0.0078.

But what if we want to store, for example, 0.0045?

It gets approximated to 0.0039, and the result loses preciseness! In this case we, as any human beings with human brains, would just say – add more bits! So they did. And of course, the number of available values can be cosmic. But there always, always will be such a value which does not 'fit' into our given bits – no matter how many of those we use. This is called quantization noise. Here come analog signals! They are not quantized by their nature – they represent *exactly* the value they have, no more, no less. Thus – you can't count it: it has infinite preciseness and no number of bits can represent it hundred percent correct. So the analog computer, in fact, measures the input parameters, operates with them and represents the results as voltage levels – which are measured by us, humans.

Of course, here I say that analog computers are *potentially* infinite-preciese: it all, in fact, depends on our measuring equipment – and it's funny to talk about some wild pinpoint results in, say, 1959, when digital voltmeters weren't a thing and voltages were measured with an arrow-based voltmeter – which usually waved its arrow every time you breathe on it. And, of course, in analog circuitry there's the counterpart of quantization noise – the electric noise that always is there, which is often also a big problem. But the analog nature of signal allows for a very special approach to task solving.

Since programming analog computers is a tough, time-consuming and – let's be honest – a pretty useless topic for most of people who would get to this paper, I'm not going to cover it here; yet for the interested ones – you're more than welcome to read this and this article, they both are very enjoyable.

Rise and decay of analog computing

It is a hard task to tell when the first analog computer was built. The technical part above touches only the electric analog computers, whereas the general approach of modeling a situation with a special machine was known and used way back before the era of electricity.

One of the most famous and mysterious ancient analog computers is the Antikythera Mechanism – found in 1902 by a group of archeologists on the greek island called Antikythera, it is debated to have been built from half a century to two centuries B.C.

The remains of the mechanism include gears and other mechanical devices: it was a purely mechanical analog computer used to predict eclipses, positions of stars and other astronomical data years in advance.

Later on, a whole lot of different analog computers were made: for example, the hydro-powered castle clock by Ismail Al-Jazari, which played music depending on the time of the day, displayed time of the year and current zodiac sign and automatically opened the doors of the castle and showed different figurines in windows depending on the time of the day, built in 11^{th} century, or a bit smaller, but way more handy thing – the astrolabe, which was used by sailors to navigate in seas.

As electricity got discovered, hybrid elecromechanic analog computers were invented around the world. As always, when humans discover a technology – they think if it can help them kill other people. So it was that time as well, with Arthur Pollen developing his Fire Control system for the navy, inspired by how inaccurate naval artillery was during training near Malta in 1900. This was a very complex hybrid electromechanic machine, mainly relying on mechanical processes – it calculated and plotted the position of the carrier ship and the enemy ship, the time which an artillery shell needs to reach from one ship to the other, and then the results were used to set the cannon direction and rise angle. This system was actively used by the Imperial Russian Navy in

World War I. In later wars, a lot of mechanic and electromechanic computers were used for artillery and missilery tasks. A number of machines was built for big companies which needed quick and precise calculations for their economic needs, and couldn't afford hiring computers (that was a name of the job back then – a person who computes)

The era of analog computers which are my main topic here – the electric analog computers – began with alternative current calculating boards, or – more simply – Network Analyzers. The need in these machines busted out in late 1920s, when a need to calculate complex AC power lines arose, and numeric calculation methods executed by humans were insufficient. These devices were literally small copies of power line setups that would be erected in future: they were simulating the power flow, stability, and other power line related parameters, but the simulation was times quicker than the process itself, and, of course, it was cheaper for electricity providers to buy a huge analog computer than to bite the bullet, wait until some extremely smart (and extremely money-charging) people complete the task, build the project and watch it explode because of some minor mistake. 50 of such machines were built, and were widely operated until late 60's. Later on, these machines got pushed out by digital computers and numeric-algorythmic problem solving, but for half of a century before that, nothing could match an analog computer in this field.

This caused an explosion of analog computers all around the world: except for (most obvious) military purposes, such as the famous Boeing B-29 heavy bomber's fire control system analog computer, they were used in civil aviation, environmental modeling, scientific researches, economy and education. Their ability to solve differential equations of high order was unbeatable by the time.

One of the great examples of such a computer is Deltar (fig. 2) Based on a computer by Johan van Veen, a netherland-based scientist, that was developed to predict tides and water flows when the geometry of the channel changes, this computer was built for a specific task – to design and implement the Delta Works is a series of construction projects in the southwest of the Netherlands to protect a large area of land around the Rhine-Meuse-Scheldt delta from the sea. The computer was used to protect a region from a natural disaster. It used very simple and exquisite parallels between water streams and parameters of electricity to operate.

"The main conceptions of the analogy useful for gaining fundamental tidal information are as follows: Direct Currents: a non-tidal river, flowing in one direction only, can be compared to an electrical conductor through which direct current is fed. Alternating Current: a tidal channel in which the ebb and flood streams go to and from resembles an electrical conductor through which alternating current pulsates. Tidal River Mouth – a channel serving the outflow of a river as well as the tide is difference in height of the water surface at two different points of a river. The term "potential gradient" means slope."
-from "Analogy between tides and AC electricity" by Johan van Veen

Deltar as of 1974

Analog computers were widely used in all spheres in need of quick calculation of very hard equations and dependencies all the way until early 80's: every big company had at least one of these machines to conduct the most crucial economical calculations, which nothing else could do. But since the making of ENIAC in 1945, digital computing wasn't frozen and abandoned at all: it was inferior to analog due to poor component base for its demands, lower quality of calculations and longer process time. But tables flipped in late 70s, when semiconductor market experienced a boom and all kinds of new transistors and other semiconductor devices appeared on daily basis, suprassing their previous models. At one point, technology advanced that much that it simply allowed to put a lot of stuff into one chip. No more tubes and bulky transistors, and things got faster. The amount of quantization noise and bit error became comparable to that of electronic analog noise in analog systems and measuring devices error; yet soon enough, it became even lower than those of analog computers. Big advantage of digital systems was the ability to store and recall programs in a way easier way than analog ones. For the latter, people would patch up a new system every time they needed to conduct a calculation. And if two consecutive calculations are needed – it's already two whole different setups, for just one complex task! Even perfocards could compete with that in terms of speed of recall, not even talking about magnetic and solid-state storage. The crossboard – a replacement for bulky patch panels – was only a theoretical project by that time: there was no such chip which would act as a voltage-controlled switch AND would be small enough to make a matrix of, say, 100 inputs and 100 outputs with such a switch on each input-output pair (10000 switches!) to make an analog computer digitally patchable. By the beginning of 80s, analog computers faded out of usage completely. Everything that was left of them is some enthusiast communities, and those very, very rare cases in science when nothing but an analog computer can do the job. This was the decay of analog computing.

Chance for an afterlife?

Nowadays, we have modern, digital computers, that recall programs immediately while remembering hundreds of those, and have levels of value preciseness so good that we don't even notice all the quantization noise in our daily lives. And everything seems to be happy and great. No one, me included, would trade that cozy PC filled with all the precious stuff in it, for a huge scary box with wires, knobs and LEDs that just solves a differential equation of some kind. But let's take the digital versus analog battle back to the field where analog was intended to be used – in science and complex computation – and see if it is that useless.

If we think of what is the most powerful digital computer nowadays, we think of a supercomputer: the return of those times when computers filled an entire room, but now it fills up an entire building. The general approach in supercomputer is, in an outline, extremely bruteforce-like and simple: "if one core does X operations per second, and two cores do more, we definitely should accumulate an astronomic number of cores under one same system and get as fast as we can." Of course the process of making this system run and be efficient is hard – otherwise there would be no world-class computer scientists involved in it. But the concept is, in my personal opinion, not the most exquisite one.

Yet it works! And it performs over 500 quadrillion (that's a lot!) floating point operations per second (FLOPS), theoretically draggable to one exaFLOPS (billon billion, 1000000000000) range. Yet it has a huge drawback: power consumption.

"Just have a look at the latest Top500-list of high-performance computers. Currently, the most powerful supercomputer is the Sunway TaihuLight at the National Supercomputing Center in Wuxi, China. This behemoth delivers a whopping 93 petaflops (one petaflop equals a quadrillion floating point operations per second), which is really unfathomable.

However, such enormous processing power comes at a cost. In this case it requires 10’649’600 processing units, so-called cores, that consume 15’371 megawatts – an amount of electricity that could power a small city of about 16.000 inhabitants based on an average energy consumption equal to that of San Francisco”
-from “Why Algorithms Suck and Analog Computers Are The Future” by Brend Ulmann

This number is stunningly huge. Imagine running a computer that eats up as much power as San Francisco? Yet, this is the reality – the faster the supercomputer is designed to be, the more energy it will consume. “Who cares?” - people say, - “if we can generate this much power we can run it”

Yes, they can. For now.

Let’s face it: world is in a big energy crisis right now. Consumption is rising exponentially while we’re still burning coal at half of electric plants. And “computers that consume more than cities” doesn’t really line up anyhow in my head with “humans not rolling planet Earth to burning dystopia”. In this situation, more exquisite decisions should be made, and a different route should be taken.

As weird as it sounds the first time, supercomputers run an OS. And that OS is Linux (of course, imagine running Windows on a supercomputer and seeing it do a BSOD?) But it doesn’t seem logical to have a multipurpose supercomputer in some cases: in a lot of them, it is used to perform the same heavy calculation over and over a lot of times. Then why not to have a machine that’s hooked up just to suit your task, yet can be ran off an average house wall outlet?

Here comes analog. A completely different approach to computing, on contrary to using algorithms, iterations and brute-force core stacking, it models a certain situation, runs the simulation and outputs the result. And nowadays, with modern measuring devices and switches, it can be just as precise as digital and patches can be stored and recalled digitally via crossboards.

Of course it wouldn’t fit for, for example, testing hash algorithms – why test a digital situation on an analog machine. But most natural things – from nuclear physics all the way through astronomy and biology, can be easily simulated on a machine driven by the same simple laws of nature: it is, in fact, stunning, how many laws are similar for different fields of science, such as biology, astronomy, electrodynamics, electronics, and so on.

I think, that the future would be much more “green” if half of the supercomputers get replaced with appropriate analog system. It just takes a fair bit of patience and knowledge in physics to create a correct simulation patch – something way too uncommon for people used to performing all the things digitally.

If the theoretical limit for supercomputer is about 1 exaflops floating point operations per second, with power consumption of approximately 165300 megawatts (derived from direct proportion of above mentioned numbers, 1000/94 petaFLOPS = X/15371 MW => x = 165279 MW), then the theoretical limit of analog computing is human brain: being a very specific analog computer, its processing power “converted” to FLOPS is estimated as 38 petaFLOPS (2.5 times lower than the above mentioned Sunway TaihuLight), but the power consumption is about 20 watts of energy: tremendous 758550 thousand times less than TaihuLight’s. And this is something to think about, I suppose!

Conclusion, references

Analog computers of the past left a rich cultural inheritance: it takes a big bit of a good scientific head to construct a fine machine, so all of that is a good point to learn from for future scientists; the approach of analog computing, the simplicity of the natural parallel between actual processes and their simulation analogs, is stunning and exquisite. There exists an analog computing fan community which explores and shares designs that anyone can build, play around and learn, as well as there are a few museums of analog computers in Europe and the US, and a lot of machines presented are still working. Another especially significant outcome of old 'classic' analog computing is the modular synthesizer scene, which experiences a second explosion nowadays: the same principle is applied into making sounds, which leads to people creating masterpieces that sound like nothing we've ever experienced before.

From the point of view of present time and future perspectives, analog computers seem to be a possible green alternative to supercomputers, and in my opinion, it might be worthy to poke this topic a bit more for computer scientists – and scientists in general, since it's a very sphere-special thing. It's definitely not over yet, and there are research teams finding out new ways and uses in analog nowadays.

"In recent years, analog computers have proven to be much more efficient at simulating biological systems than digital computers. But existing analog computers have to be programmed by hand, a complex process that would be prohibitively time consuming for large-scale simulations.

Last week, at the Association for Computing Machinery's conference on Programming Language Design and Implementation, researchers at MIT's Computer Science and Artificial Intelligence Laboratory and Dartmouth College presented a new compiler for analog computers, a program that translates between high-level instructions written in a language intelligible to humans and the low-level *specifications of circuit connections in an analog computer.*

The work could help pave the way to highly efficient, highly accurate analog simulations of entire organs, if not organisms."

-from "Analog Computing Returns" by Larry Hardesty, MIT news, June 2016

List of sources:

Why Algorithms Suck And Analog Computers Are The Future, by Bernd Ulmann

Analogy between Tides and Electricity A.C., by Dr. Johan van Veen for The Engineer, 1947

Analog Computing Returns, by Larry Hardesy for MIT News, June 2016

Soulless Mathematician: history of computing devices before PCs (ru), by Pavel Shubetsky for Igromania, 2009

Gears of War: when mechanical analog computers ruled the waves, by Sean Gallagher, March 2014

Analog and Hybrid Computers @ Technikum29 living museum

Page of analog museum in Germany, big inspiration