Digital Studies / Le champ numérique

Aljayyousi, Mohammad Ibrahim. 2022. "The Computational Fallacy: A New Model for Understanding the Role of Computers in Humanities." *Digital Studies/Le champ numérique* 12(1):15, pp. 1–19. DOI: https://doi.org/10.16995/dscn.8105

Open Library of Humanities

The Computational Fallacy: A New Model for Understanding the Role of Computers in Humanities

Mohammad Ibrahim Aljayyousi, Philadelphia University, JO, mohammad.aljayyousi@gmail.com

The paper tries to counter some misassumptions about the computer and computation especially in relation to humanities and human behavior and they amount to what the author calls "the computational fallacy. The paper discusses a number of points such as the physiology of the computer, the Etymology of basic terms in the field, and the current approaches especially in mainstream digital humanities which reduce computation to a set of "tools". The paper then proceeds to discuss some counter arguments such as rethinking the notion of programmability which should substitute "calculation" as the core of computation, considering the transformative nature of the computer and its media using the ideas of some theorists like Manovich and Drucker, and some new approaches that view computation differently like Computational Thinking, Algorithmic criticism, and Speculative computing.

Cet article essaie de contrer quelques suppositions erronées sur l'ordinateur et la computation, plus particulièrement leur relation aux sciences humaines et au comportement humain qui équivaut à ce que l'auteur appelle « the computational fallacy » ou l'erreur computationnelle. Cet article aborde de nombreux points, tels que la physiologie de l'ordinateur, l'étymologie de termes de base dans ce domaine, ainsi que les approches courantes, particulièrement les approches dominantes dans les humanités numériques qui réduisent la computation comme un ensemble « d'outils ». Ensuite, l'article poursuit en discutant les contre-arguments comme les nouvelles réflexions des notions de programmation qui devraient substituer le calcul au cœur de la computation, considérant la nature transformante de l'ordinateur et ses médias utilisant les idées de quelques théoristes dont Manovich et Drucker, et quelques nouvelles approches qui voient la computation différemment comme « Computational Thinking » ou la pensée computationnelle, « Algorithmic criticism » ou la critique algorithmique et « Speculative computing » ou la computation spéculative.

1.1 Background and definition of the problem

Despite the consensus about a digital revolution underway, there is a group of established misconceptions about computing and the computer which constitute what I would like to call "the computational fallacy." This fallacy downplays the radical nature of the digital revolution and re-establishes the computer as just another tool or machine. It also means the literal understanding of computation. Ironically, the basic claims are hard to deny because they are based on facts, such as the genesis and the physics of the computer. But stopping here creates a partial view that brackets off other significant facts, like the roles and diverse functions that the computer performs in real life. Such a view is also blind to the many ways the computer is able to transform, or even evolve, into a spectrum of things when it is put to use, almost all the time in an interactive environment with humans. The computer is a machine, but a full stop does not follow. In fact, the computer started with one of the most ambitious and challenging projects of science, a "thinking machine." This was the vision of Alan Turing (1950), the founder of computer science. The computer is not exactly like a calculator, a TV, or even a data processor in the general sense, even though it efficiently does some functions associated with these machineries.

Perhaps the physiology of the computer is so imposing and self-evident. The typical image of a pc or laptop with a screen, keyboard, and a mouse leaves no room for imagination. Moreover, the ability of the computer to embed old media makes it easy for it to pass as an advanced multimedia machine, or a super-TV. This "façade" entangles the computer in the machine corporeality. The computer screen, for example, is bequeathed from earlier technology; the cinema and the TV are the closest relatives. However, it does not exactly work in the same way as a display tool designed for a passive viewer. Because the computer depends on continuous input from the user, its screen serves as a channel of output and input, and an interface between the user and the machine. Besides, the content that the screen "displays" or visualizes is much more diverse; we see all types of media, and there are also other types of content peculiar to the computer and its mechanism of work. When a programmer sits at the screen, he or she is actually "seeing into the mind" of the computer. The screen is also a platform of work for the user. We create and make things happen on the screen.

We agree that the computer, unlike other machines, performs a special set of functions that are easily aligned with human thinking processes, hence the qualification as an intelligent machine. The computer does this in interaction with a human user, and with "nontrivial" effort on the user's part most of the time. This should sound simple and straightforward. However, because of the workings of the computational fallacy, this interactive dynamic is usually marginalized, and we tend to think of the

computer as working independently. The role of the computer's user goes beyond turning on and off the machine or providing input, and this role is further "amplified" by a larger set of input devices like keyboard, mouse, etc. The interactivity of the user, and we are talking about the ordinary user here, is consistent and continuous. Besides, there is another set of specialized or expert users, like programmers and designers, whose interaction is tantamount to a form of communication with the computer. The outreaching significance of this aspect called for a whole new field of research, which is Human-Computer Interaction, HCI.

The tendency to envision the computer as working independently is related to a desire in the relation between man and machine, emphasized by the industrial revolution, to seek automation at all costs. What this means is that the purpose of any machine is the total removal of human effort. This objective is usually centralized and used as a standard for the machine's performance. This view is obviously mistaken when it comes to computers. As the previous discussion shows, the human-computer interactivity is an integral part of the machine's functioning, and the need for human effort is not simply a disadvantage in this particular case. Moreover, it is not difficult to argue that removing human effort is ridiculous when it comes at the expense of performance quality, and even if we use the same standard, we can see that regarding many operations done on the computer, human effort is radically transformed in nature and tremendously minimized in amount.

The computational fallacy does not necessarily exist as a unified theory or approach. Rather, it is evident as a set of assumptions and underlying beliefs that form part of the public imagination and are also implicit in many theories and critical paradigms. By suggesting a unified term for this phenomenon, I intend to give a systematic articulation of its origins and claims, which precedes my attempt at refuting those claims.

1.2 Etymology

Let us look at the origins of this fallacy. Some etymological and historical notes would help here. The Merriam-Webster dictionary (2021) provides the following meanings for the entry "compute": "to determine especially by mathematical means," "to make calculation," and to "reckon." All of these meanings were also part of the original Latin word "computare" from which "compute" is derived. The absence of any semantic distance between the word and its origin is in itself significant. It seems that computing is a static notion that defies the passage of time, or more accurately, it is a concept that belongs to the basic science, the protoscience, which is mathematics. Mathematical calculation is the core meaning of computing, and even the one sense that seems to depart a little, "reckon," refers to reckoning that comes as a result of calculation,

again by mathematical means. The etymological line is unbroken, and the semantic boundaries are so rigid. On the cultural and epistemological level, this shows the hegemony of mathesis and its cultural authority with all the connotations of accuracy, systematicity, empiricism, and disambiguation, a topic that we will be returning to. Suffice it to point for the time being that the computational fallacy seems to originate here.

1.3 Programmability: The ignored breakthrough

A look at the early history of computing tells us that the computer evolved from automatic calculators. Automatic calculation is still central to modern digital computers, which remain fundamentally sophisticated calculators. However, besides automatic calculation, there was another technology that led to the birth of computing as we know it now and, in fact, made all the difference. It is by virtue of this technology that the computer has its cultural authority in contemporary life. This was the technology of programmability. Interestingly enough, this magnificent technology did not rest upon or come as a result of any material innovation beyond automatic calculation itself. It was simply a smart manipulation of the very method and process of automatic calculation that had then been enhanced by powerful "computers." Programmability is so simple yet so creative. The machine starts to "behave" in a desired way. Programmability is basically the process of making the computer able to receive instructions, basically by an orchestration of automatic calculations. In doing so, programmability "links automated processing to the symbolic realm" and adds "an extra level of articulation in coding symbolic values onto binary entities that could be processed electronically," as Johanna Drucker says, and this "made the leap from automated calculation to computation possible" (Drucker 2009, 23). Therefore, computation as we know it today is synonymous with programmability, which was indeed a great leap not only towards enormous improvement of functionality, but also towards a transformation of automatic calculation itself and its uses.

Of course, programmability in its basic sense was not new. Machines were programmed to operate in a certain way long before computers using mechanical methods most of the time and in a restricted way. By contrast, computational programmability is integrated within the mechanism of automatic calculators, so its uses are never exhausted. Programmability upgraded computation to real communication with the machine. It has its own languages and language-like method, which is code. In fact, programming languages are correctly called so. Not only are they syntax-based sign systems but, more importantly, they serve as effective communication tools, and even augmented tools compared to natural languages, if we consider "the multiple addressees … which

include intelligent machines as well as humans" (Hayles 2005, 15). This multiplicity of addresses is why code is code. It works on at least two levels of interpretation linking human and machine epistemologies. The computer code translates between the human and the machine.

Programming and code are central to the present topic as they help illustrate the confusion about computing that the computational fallacy reflects. With programmability, automatic calculation is transformed and moved to a different epistemological plane, that of instructions and behaviour. The process of calculation is not the end in itself but just a means. This bridges the gap between human intentionality and machine automation. The great and complex leap entailed here implicated the machine and its user, the human, in a new dynamic of interaction and marked a new phase for the machine. The machine would once and for all transcend its forefathers and be connected forever to humans in a complex network of functionality that was soon to invade all aspects of life. The cultural authority of code started here, by virtue of this leap. One way of coming to terms with the computational fallacy is to see it as the blindness to the epistemological nature of the transformation that programmability introduced to computing. The computational fallacy is also blinded by the fundamentality of automatic calculation that remains central to the whole process. In short, the computational fallacy blurs the line between automatic calculation and programmability, insisting that the latter has not provided any major break with the earlier.

1.4 New media and the transformative nature of the computer

Through programmability and another feature we are going to talk about, emergence, the computer has achieved universality, perhaps not in the sense that is meant by Turing (1950), but in the sense of the computer's ability to embed other machines and media, which explains its ubiquitousness. Digitization is a process that reflects this ability of the computer. When the computer "mediates" the material processed on it, there is a metamorphosis underlying this process. This is why computer mediation popped up naturally as an established concept in the discourse about computation. As a result of digitization and computer mediation, we have a totally new category of media, called new media. New media is a case in point. In his *The Language of New Media*, Lev Manovich specifies five principles of new media which are supposed to account for the newness of this media (Manovich 2001).

Manovich mentions the following principles: numerical representation, which refers to the fact that "a new media object can be described formally (mathematically) and "is subject to algorithmic manipulation" (Manovich 2001, 27); modularity, which

means that new media objects are "represented as collections of discrete samples" (Manovich 2001, 30); automation; variability; and transcoding. Manovich explains the last principle by the following note:

> [T]he logic of a computer can be expected to significantly influence the traditional cultural logic of media, that is, we may expect that the computer layer will affect the cultural layer. (Manovich 2001, 30)

These principles overlap with the work of other theorists who also provide similar principles of computational media, not necessarily in the general sense used by Manovich. Some concentrated on new forms of textuality mediated by the computer. Katherine N. Hayles, for example, talks about four characteristics of digital or computer-mediated text. The first two characteristics she provides are layeredness and multimodality (combining text, image, and sound together). Hayles (2008) adds that storage in computer-mediated text is separate from performance unlike print books for example where the same artifact functions as both storage and performance mediums. The last characteristic she mentions is that digital text manifests fractured temporality (Hayles 2008, 164–165). It is not hard to see that both Hayles and Manovich are describing computer mediation as such.

The computer is never a passive or "harmless" medium for transmission. The five principles and the four characteristics describe some direct consequences of computer mediation, which indeed amount to a separate "logic" or layer, in Manovich's terms, that affects everything else. Taken together, these principles indicate the emergent working of the computer, and this is something that Manovich does not fail to notice:

> [B]eginning with the basic, "material" principles of new media—numeric coding and modular organization—we moved to more "deep" and far-reaching ones—automation and variability. (Manovich 2001, 45)

Thus, we can talk about two separate manifestations of computation, divided by an epistemological line. We have the basic ones, the mathematical or material, on the one hand, and the emergent, deep ones on the other. Depth is perhaps not the best choice of words, but the notion is complicated enough. I prefer emergence because this is the notion that can explain the relation between these two sets of principles/levels. The different levels are "genetically" related while at the same time they are distinct. The line dividing the two shows an epistemological leap as a one-to-one correspondence

is obviously lacking. Again here, we say that dwelling on the basic level points to the computational fallacy, which is blind to the implications of the epistemological leap. Transcoding is perhaps the principle that is supposed to summarize the overall imprint of the computer:

> [I]n short, what can be called the computer's ontology, epistemology, and pragmatics—influence the cultural layer of new media, its organization, its emerging genres, its contents. (Manovich 2001, 46)

What needs to be stressed, regardless of the controversy about the reaching of the influence designated here, is that computer mediation has strong ontological and epistemological implications. The computational fallacy is the denial of this fact or the failure to recognize it.

But recognizing, and then acknowledging, this transforming influence of the computer is not without difficulty because of the way computer mediation itself works. Perhaps a parallel question to the one about the newness of new media is why it is media, or what is in it that makes it retain the status of media. Computer-mediated media is media by another name; thus, we talk about digital film, digital photography, digital poetry, etc. This is because the computer and other mediums "intermediate" or "remediate" each other, as Katherine N. Hayles and David Bolter and Richard Grusin call this respectively. This indicates "complex transactions" between different forms of media (Hayles 2005, 7), and as Bolter and Grusin put it, it "involves both homage and rivalry, for the new medium imitates some features of the older medium, but also makes an implicit or explicit claim to improve on the older one" (Bolter and Grusin 2000, 23). We have seen how the computer "improves" on the material processed on it, but it is equally important to see how it "imitates" that same material, allowing it to retain its original medial categorization. The computer can simulate; it creates a simulacral "façade" and a surrogate physicality of/for the original media. I call it a façade because it reveals just the tip of the iceberg and hides all the inner workings. We tend to be blinded by this due to assumptions about materiality, representation, and originality.

In this way, the computer appears to act so smartly. It meets the requirements of the dominant culture in terms of mimesis and reproduction while hiding its transformations away from the skeptical eye of that culture. Or we can say it makes a trade-off. Without drifting into sci-fi scenarios, it is in this sense that the machine is actually taking over. Those of us who insist that the computer is just a medium for transmission abiding by whatever rules we assign are in fact fooled. This is one form of the computational fallacy, the insistence that the machine is just reproducing old media for us.

1.5 Emergence

Another aspect of computation and the computer that is worth noticing is its emergent way of working. The idea of emergence is not new, and it has its philosophical and scientific applications. We can talk about emergence when things can be understood as multi-level complex systems where relatively simple patterns lead to emergent more complex ones. O'Connor provides the following definition and short history of the term:

> Emergence is a notorious philosophical term of art. A variety of theorists have appropriated it for their purposes ever since George Henry Lewes gave it a philosophical sense in his 1875 *Problems of Life and Mind*. We might roughly characterize the shared meaning thus: emergent entities (properties or substances) "arise" out of more fundamental entities and yet are "novel" or "irreducible" with respect to them. (O'Connor 2020)

In short, emergence is the transformation from fundamental to novel where the novel is irreducible or, for that matter, untraceable in any obvious way to its fundamental origins. This point is significant to our argument about the computational fallacy since this fallacy simply occurs when this emergent nature of the computer or the link between the fundamental level, which is computation, and the many emergent levels is not recognized or acknowledged.

Emergence and computation are so tightly related that they become synonymous terms in the work of emergence theorists. Some of these, as Hayles (2005) points out, universalize the computational–emergent paradigm:

> Recently, however, strong claims have been made for digital algorithms as the language of nature itself. If, as Stephen Wolfram, Edward Fredkin, and Harold Morowitz maintain, the universe is fundamentally computational, code is elevated to the lingua franca not only of computers but all physical nature. (Hayles 2005, 15)

They also understand the universe "as software running on the 'Universal Computer' we call reality" (Hayles 2005, 27). The computational fallacy ignores the overreaching implications of this emergent mechanism of the computer and focuses on the basic level that is "irreducibly" mathematical. The emergence in the computer also means that its uses are hardly exhaustive, and remain open for new, never previously imagined, manipulations.

Computation starts with "computation" but ends up with numberless "emergent" patterns, some of which are not computational in any obvious sense. The computer works on different levels of complexity. On one level, we have machine language or code, and this is the "brute" pattern of "naked" digitality, or bits (ones and zeroes). All computational operations are physically carried out at this level. Many complex levels are based on this basic one and might be said to actually emerge out of it—for example, the level on which programmers work with higher codes like C++ or Java, or the GUI level through which ordinary users interact with the computer. The computational level, in its mathematical rendering, serves as a basic level on which more complex, more advanced levels and functions "emerge."

Let us take an illustrative example from a common feature, the digital photo. As represented and therefore shown on the screen of the computer, a digital photo is perceived as a flower. However, if we enlarge any bit of it, we can see a pixelated section consisting of coloured pixels. All of us, the experts and the novice, know that a pixel is a tiny square on the screen. This is the upper layer (the novel) of the digital space. As stored in the computer's memory, the digital photo and the rather complex arrangement of the colour pixels are only digits. Those digits are the fundamental layer on which the photo, as complex as it seems, is based. Another less obvious example is the interface of the operating system like Windows. The user-friendly and interactive interface that revolutionized computing is "the façade" of deeper layers. We as users interact with Windows and perform simple tasks like opening a program, giving a print order, editing a photo, etc. All this happens on the visualized level, that of the user interface. The operating system is programmed or written in a language more complex that the one the user interacts with, usually visual C or C++, which itself is based on the assembly language, which, in turn, is based on the digital duality of zero/one or electricity/no electricity. Those levels of code or "language" are based on an emergent hierarchy; each level builds a higher one that gives sophisticated results.

Finn in *What Do Algorithms Want?* refers indirectly to emergence as it pertains to computation in the following remark:

> By assembling systems that follow a few simple computational laws, we can iterate toward highly sophisticated solutions to difficult problems that resist more straightforward (e.g., human-designed) algorithmic approaches. (Finn 2017, 183)

The "simple" computational processes can allow more sophisticated ones to emerge, and this emergence is not final in its form; it is open to more sophistication.

1.6 The discursive presence of the computational fallacy and sample endeavours to counter it

Let us move into more specialization and see how the computational fallacy manifests itself in the humanities discourse. Some of the previous claims are easy to pop up when computation or the computer is in communication with an alien, or supposedly so, register like the humanities. A central issue here is the seemingly enduring disparity, disjunction, or incompatibility between the literary enterprise and the computational method. Computation is quantitative, digital, rule-governed, mathematical, accumulative, data-driven, algorithmic, in short, scientific, while literary study is qualitative, analogue, subjective, interpretative, intuitive, serendipitous, in short, humanistic. We are faced with a classical dichotomy. The disparity here, which is between two separate domains, is not a problem in itself. After all, human knowledge is compartmentalized into fields that, by definition, do not share common epistemological grounds. But with the increasing "fault lines" between the two fields or registers, basically with the advent of digital technology, we are forced to rework and refocus this dichotomy. We find ourselves in search for a compromise for what has become a de facto "clash" between our two major epistemological tools:

> Much of the intellectual charge of digital humanities has come from the confrontation between the seemingly ambiguous nature of imaginative artifacts and the requirements for formal dis-ambiguation essential for data structures and schema. (Drucker and Nowviskie 2004)

This confrontation is a multi-faceted one, and it has raised a set of epistemological questions.

It is logical to assume that the very idea behind digital literary studies and digital humanities in general entails a contention with and a rejection of the computational fallacy. Ironically, this is not the case, and the working of this fallacy is evident and represents a major drawback in the field as I am going to show. The main issue remains the fundamental disjunction between computation and literary studies, which is the central claim of this fallacy. This issue is already centralized, and we can easily consider all scholarship in digital humanities and its offshoot, digital literary studies, as contentions with this basic question. The answers, however, are still guided and parameterized by the computational fallacy.

1.6.1 Algorithmic Criticism

Stephen Ramsay's "Algorithmic Criticism" includes an insightful analysis in this regard. In an attempt to rethink the conditions for re-integrating the algorithmic manipulation

of literary texts into literary studies, Ramsay (2013) points to a number of key factors. The bottom line is that computing and its corresponding digital revolution have not "penetrated the core activity of literary studies which … remains mostly concerned with the interpretive analysis of written cultural artifacts" (Ramsay 2013). The implicit assumption here is that any possible "penetration" by computation means a radical change of that core activity, which is inherently resistant to computation. The problem is that the available computational tools are still behind in terms of appropriating this core activity, which is irreducibly subjective and is based on a different hermeneutical model or rubric in which "the accumulation of verified, falsifiable facts" is the basis of interpretation (Ramsay 2013). We lack the "tools that can adjudicate the hermeneutical parameters of human reading experiences—tools that can tell you whether an interpretation is permissible," and they still "stretch considerably beyond the most ambitious fantasies of artificial intelligence" (Ramsay 2013). The subtext of this description, which is an accurate and faithful one as long as the original assumptions it starts from are concerned, is the fundamental disjunction between the two parties, computing and criticism, and this creates a perfect stalemate.

Ramsay (2013) finds a way out of this stalemate in the context of algorithmic criticism. Although the transformation allowed by the algorithmic manipulation of literary texts cannot be intractably linked to the type of interpretive conclusions that we seek in literary studies, he affirms, it "can provide the alternative visions that give rise to such readings" (Ramsay 2013). We can still use textual analysis because any interpretation involves a radical transformation of texts; therefore, "the narrowing constraints of computational logic—the irreducible tendency of the computer toward enumeration, measurement, and verification—are fully compatible with the goals of criticism" (Ramsay 2013).

However, a different conclusion is possible if that bottom line is rethought. Let us agree that the facts about criticism are hard to question, and they are backed by established disciplinary and epistemological boundaries. Besides, as Ramsay (2013) points out, any radical change in criticism and its core activity would make it cease to be criticism. The scene is so different regarding the other party, computing and the computer. The disciplinary boundaries, if any, are less rigid, and the epistemological and theoretical parameters governing the field are still open to discussion. I suggest a new understanding outside the constraints of the computational fallacy and its givens about the computer. A different kind of conclusion than the one reached by Ramsay becomes inevitable.

1.6.2 Speculative computing

Another case in point comes from the work of Johanna Drucker and her Speculative Computing, the approach that she theorizes in *SpecLab: Digital Aesthetics and Projects*

in Speculative Computing. Speculative Computing, whose abbreviation SC reverses that of CS (computer science)—not a totally insignificant thing, especially as speculation replaces science—is a pertinent example for many reasons; it is a fully fledged theory and a self-conscious attempt at an alternative approach in digital humanities, one that presents a humanistic appropriation of computing, in addition to the fact that it links theory and practice (SpecLab projects). The starting point of this approach is "a serious critique of the mechanistic, entity-driven approach to knowledge that is based on a distinction between subject and object" (Drucker 2009, 21). The name given to this alternative theory is aesthesis, which is "a theory of partial, situated, and subjective knowledge" (Drucker 2009, xiii). Aesthesis is meant to contrast with and counter mathesis, which represents the mechanistic approach with all the implications of this.

Another starting premise of SC is a self-conscious awareness of the computation/humanities epistemological friction:

> The humanistic impulse which "has been strong in its dialogue with "informatics" and "computing" but has largely conformed to the agenda-setting requirements set by computational environments. Our goal at SpecLab, by contrast, has been to push against the logical constraints imposed by digital media. (Drucker 2009, 22)

These "agenda-setting" requirements are logical systematicity, formal logic, and disambiguation, as Drucker points out at different places, and are all patently counter-humanistic. The use of the generalist term "computational environments" is also significant, and I will return to this later. SC also presents an alternative mechanism of work within these environments:

> We used the computer to create aesthetic provocations—visual, verbal, textual results that were surprising and unpredictable. Most importantly, we inscribed subjectivity, the basis of any interpretive and expressive representation into digital environments by designing projects that showed inflection, the marked specificity of individual voice and expression, and point of view as a place within a system. (Drucker 2009, 19)

We see how theory and practice are entwined. In fact, they are inseparable. The theoretical agenda of inscribing the humanistic is translated into design projects in SpecLab. This means specific decisions on the techno-practical level. Let us take two examples of such decisions from one SpecLab project, which is *Temporal Modelling*:

> [O]n a technical level, the challenge is to change the sequence of events through which the process of "dis-ambiguation" occurs. Interpretation of subjective activity can be formalized *concurrent with* its production—at least, that is the design principle we have used as the basis of *Temporal Modelling*. (Drucker and Nowviskie 2004)

Changing the sequence of events means shifting the epistemological prioritization, or as put by Bethany Nowviskie, Drucker's collaborator: "the subjective, intuitive interpretation is captured and then formalized into a structured data scheme, rather than the other way around" (Drucker and Nowviskie 2004). The importance of this example, besides specificity, is that we have an idea about how SC works; the rules of the game are changed within the computational environment. In doing so, SC realizes its premise in contrast to dominant practices in DH:

> The digital humanities community has been concerned with the creation of digital tools in humanities context. The emphasis in speculative computing is instead the production of humanities tools in digital contexts. (Drucker 2009, 25)

Projects in SC are not just technical experiments but have "ideological as well as epistemological" aims. Ideologically, the ultimate aim, as declared by Drucker (2009), is "to push back on the cultural authority by which computational methods instrumentalize their effects across many disciplines." The target, especially in relation to computing, is further clarified:

> The villain, if such a simplistic character must be brought on stage, is not formal logic or computational protocols, but the way the terms of such operations about administration and management of cultural and imaginative life based on the presumption of objectivity. (Drucker 2009, 5)

This clarification is never redundant and very crucial. The problem is in the administration, which "locks computing into engineering problem-solving logical sensibility" (Drucker and Nowviskie 2004). The relation between logical systematicity and computing is rethought and the assumed synonymity is broken; this really amounts to a revelation. Computing had been exclusively used for logical ends, but this was due to the lack of alternative approaches:

> We know, of course, that the logic of computing methods does not in any way preclude their being used for illogical ends—or for the processing of information that

> is unsystematic, silly, trivial, or in any other way outside the bounds of logical function. (Drucker and Nowviskie 2004)

SC and its corresponding SpecLab projects have thus provided a positive answer to most of the questions that they set out to address:

> Can speculation engage these formalized models of human imagination at the level of computational processing? … What might those outcomes look like and suggest to the humanities scholar engaged with the use of digital tools? Does the computer have the capacity to generate a provocative aesthetic artifact? (Drucker and Nowviskie 2004)

The computer definitely has a generative aesthetic capacity, not because this is an inherent capacity in it, but rather, because the computer's main capacity is in being adaptable and susceptible to different uses. The question is to have a framework and a technical blueprint; this is what the theorists of SC have done.

SC is a lesson that we need to learn. The computational fallacy is the insistence to lag in the theoretical and ideological atmosphere that SC has rendered incongruous. It is the insistence that the villain is computing itself, independent of the approach in which it is appropriated. The computational fallacy is using an instrumentalist approach while maintaining that the computer is an irreducible instrument. Computing provides an environment; this is why I noted the use of this term. The word "environment" is a good choice as it indicates that computing provides surrounding conditions rather than imposes any specific course of events.

1.6.3 Experimental Humanities and Machine Learning

Ed Finn in *What Do Algorithms Want?* introduces the term "Experimental Humanities" as a redefinition of roles between humanities and computation, represented in the algorithm:

> We can choose to construe the figure of the algorithm as a god to be worshipped … or we can choose to see a new player, collaborator, and interlocutor in our cultural games. This is what I would like to call "experimental humanities." (Finn 2017, 192)

The algorithm, as Finn shows, is not only becoming a central player in the cultural games, but our culture now depends on a set of "culture machines" or digital tools that take "a growing share of the critical and creative work that used to be distinctively,

intrinsically human" (Finn 2017, 181). They "collaborate" with us on activities and domains that we had long thought were exclusive to human intelligence.

Finn's proposition is a powerful one and it has affinities with my attempt in this article to counter what I have called the "computational fallacy." Experimental criticism with its acknowledgment of the more powerful role that algorithms and computation have in our cultural production, and not to approach them merely as mediums of transmission. Qualifying this field as experimental is also important since the prospects are never predefined or delimited. All are open to new possibilities. The computational fallacy is the attempt to counter this trend.

Machine learning, especially as it pertains to creativity, is valid for our current discussion. The field is advancing and shifting in approaches and paradigms so rapidly that a year seems like a century compared to other disciplines. Some studies worth mentioning are Tegmark (2017); Miller (2019); and Foster (2019). Engaging with the most recent theories or applications in detail is beyond the scope of this article. However, another supporting log to the claims which have been proposed against the computational fallacy can be found in this field. A major one is the need to continuously redefine and therefore demarcate the epistemological boundaries of fields and disciplines and their core concepts. Miller (2019) and Tegmark (2017), for example, start and also end with redefinitions of well-established concepts such as intelligence, creativity, consciousness, and, for that matter, humanness. The counter-fallacy evidence we can take from this is that what is computational and what is humanistic are hard to establish. It is even counterproductive to try to set rigid boundaries between the two fields and see them as intrinsically incompatible or even compatible. Machine learning has shown us that they are becoming hybrid.

1.7 Conclusive remarks

There is a question that might easily go unasked. Why is it computing and computational environments that are in question here? The larger ideological framework for SC, which is aesthesis, is a critique of the instrumental logic in western culture and not exclusive to digital humanities or practices generally related to computing. It is either that computation, especially in humanities context, provides an extreme case of the working of the instrumental logic, or it serves as a perfect environment for countering that logic and "demonstrating" the alternative approach: it allows the entanglement of theory and practice, for example. If we disagree about the earlier, we can easily agree on the later, and supportive examples from SC itself abound. Perhaps it is useful to conclude here with a reminder from Drucker herself:

> No task of information management is without its theoretical subtext, just as no act of instrumental application is without its ideological aspects. We know that the "technical" tasks we perform are themselves acts of interpretation. (Drucker and Nowviskie 2004)

So, we can in no way continue investing the computer with the vices of logical systematicity, declaring it forever disjunct with humanities. The dominant trends in computing cannot be separated from their theoretical and ideological subtext. Moreover, and more importantly, any change in the subtext will result in a radical change in the nature of information management, which means, among many other things, that there is a great potential in terms of alternative approaches, in the manner of SC.

If we start from a different view of computation, one that is free of the computational fallacy, a whole set of new conclusions will be available. Computation's disjunction with literary studies is not a given. A different view means reworking this disjunction, leading to new terms of engagement between the two fields. A number of key issues will need to be revisited. Computation is not only a set of tools or methods, not to mention an irreducible set. Similarly, the claim that the only available role for computation in criticism is textual analysis is acting like a wall or a red line. All these assumptions lead to the tendency to frame the problem in terms of technical limitations and maintain the premise that the problem is that CS and AI are still behind regarding human-like abilities. The claim to technical limitations historicizes the disjunctive moment and implies that current uses are the best available ones. Another outcome of this type of thinking is that the issue remains within the science/humanities dichotomy, with computation as a scientific (and scientifying) method, and digital literary studies easily becomes another attempt at scientifying the humanities.

But the question "What does it mean to think outside the computational fallacy, especially as pertinent to literary studies?" remains pending. Answering this question is beyond this article, in addition to the fact that the answers depend on potential work that marries theory and practice, much like speculative computing. However, I can presently suggest one direction in this regard. One of the aspects of the new understanding is envisioning the computer as a partner. Acknowledging partnership would mean a division of labour between man and machine: who can do which better. The computer is not supposed to work independently but in a self-consciously interactive environment that allows both parties (human and computer) to share and exchange information and insight. Successes and failures would thus be judged from the same collaborative perspective. Computational artifacts and interfaces already have many

tools for interaction, like the pop-up message windows, by which "the machine" gets input from the human user. Those can be built upon for a more effective and integrated interaction.

Let us take the example of machine translation. If we look at the current software for machine translation, we will notice the limiting influence of the automation-at-all-costs requirement that I talked about earlier. The machine is "left alone" to do the tasks, and the human's role is to provide the input, usually by a simple procedure like copy-paste. An alternative and definitely more efficient model would be a collaborative one, which would include tools that enable the program to ask for feedback from the user on challenging issues, usually those that cause faulty translations, and this should be throughout the translation process.

The collaborative model allows a new understanding of the role of computation within any humanistic practice. By virtue of this model, we are able to see that any incompatibility is due to the assumption that the computational device is supposed to work independently. If we start to think of it as a full partner, our whole approach will change, and incompatibility is no longer an issue. All in all, a collaborative model will definitely alter the epistemological underpinnings of literary studies because the logistics of truth assignment in the field will radically change. When analysis or interpretation is done with the participation of a machine, problems will be reformulated not only for human reasoning or intuition but for a machine's "intelligence," basically in an algorithmic manner. It is in this way that literary studies might be said to be "scientified." There are two interrelated sides to the CF: the literal understanding of computation, and the understanding of computation as inherently and exclusively "scientific." As this article has hopefully shown, computation is an environment for work, and it is open to all types of agendas and subtexts. This is an advantage unless we start from an entrenched ideological position that insists on one certain appropriation of computation.

Competing interests

The author has no competing interests to declare.

Editorial contributions

Recommending Editor
Gimena del Rio Riande, CONICET, Argentina

Section Editor
Iftekhar Khalid, The Journal Incubator, University of Lethbridge, Canada

Bibliography Manager
Shahina Parvin, The Journal Incubator, University of Lethbridge, Canada

Copy and Layout Editor
Christa Avram, The Journal Incubator, University of Lethbridge, Canada

References

Bolter, David, and Richard Grusin. 2000. *Remediation: Understanding New Media*. The Cambridge: The MIT Press.

"Compute." 2021. Merriam-Webster Online Dictionary. "Compute." Accessed October 11. http://www.merriam-webster.com/dictionary/compute.

Drucker, Johanna. 2009. *SpecLab: Digital Aesthetics and Projects in Speculative Computing*. Chicago: University of Chicago Press. DOI: https://doi.org/10.7208/chicago/9780226165097.003.0001

Drucker, Johanna, and Bethany Nowviskie. 2004. "Speculative Computing: Aesthetic Provocations in Humanities Computing." In *A Companion to Digital Humanities*, edited by Susan Schreibman, Ray Siemens, and John Unsworth. Oxford: Blackwell, 2004. Accessed October 11, 2021. http://www.digitalhumanities.org/companion/view?docId=blackwell/9781405103213/9781405103213.xml&chunk.id=ss1-4-10.

Finn, Ed. 2017. *What Do Algorithms Want? Imagination in the Age of Computing*. Cambridge: The MIT Press. DOI: https://doi.org/10.7551/mitpress/9780262035927.001.0001

Foster, David. 2019. *Generative Deep Learning*. Newton, MA: O'Reilly.

Hayles, Katherine N. 2005. *My Mother Was a Computer: Digital Subjects and Literary Texts*. Chicago: University of Chicago Press. DOI: https://doi.org/10.7208/chicago/9780226321493.001.0001

———. 2008. *Electronic Literature: New Horizons for the Literary*. Notre Dame: The University of Notre Dame Press.

Manovich, Lev. 2001. *The Language of New Media*. Cambridge: The MIT Press.

Miller, Arthur I. 2019. *The Artist in the Machine*. Cambridge: The MIT Press.

O'Connor, Timothy. 2020. "Emergent Properties." In *The Stanford Encyclopedia of Philosophy*, edited by Edward N. Zalta. Accessed October 11, 2021. https://plato.stanford.edu/archives/fall2020/entries/properties-emergent.

Ramsay, Stephen. 2013. “Algorithmic Criticism.” In *A Companion to Digital Literary Studies*, 477–491. John Wiley & Sons, Ltd. Accessed October 11, 2021. DOI: https://doi.org/10.1002/9781405177504.ch26

Tegmark, Max. 2017. *Life 3.0: Being Human in the Age of Artificial Intelligence*. New York: Knopf.

Turing, Alan. 1950. “Computing Machinery and Intelligence,” *Mind*, LIX (236): 433–460. Accessed October 11, 2021. DOI: https://doi.org/10.1093/mind/LIX.236.433