Paper TT06

Red Pill or Blue Pill? Assessing the Impact of Generative AI on Pharmaceutical Programming

Paul Shannon, Katalyze Data, Oxfordshire, UK

ABSTRACT

Recent Artificial Intelligence (AI) services such as OpenAI GPT-4 have received wide media attention, being reported as gamechangers in the use of AI for many tasks, including automation of computer programming. Are such AI technologies posing an existential threat to the traditional programmer, and what risks are involved in adopting any such technology in a heavily regulated industry such as pharmaceutical research?

There are previous examples of what seemed like seismic changes which offer clues to how AI will ultimately change the programming industry. This paper reviews how past technological advancements has affected industry trends, where technological possibility conflicts with responsibility and accountability, assess AI's ability to adhere to industry standards like CDISC or company-specific coding standards and frameworks, and consider the likely real-world impact on changes to the industry of pharmaceutical programming.

INTRODUCTION

Generative AI (Gen-AI) has been the hottest topic in the last year (reference 1). Most if not all attendees at this year's PhUSE conference will have heard of it already and many will have discussed changes in your workplaces that can be brought by Gen-AI; and if you haven't, you will almost certainly change this after attending this conference.

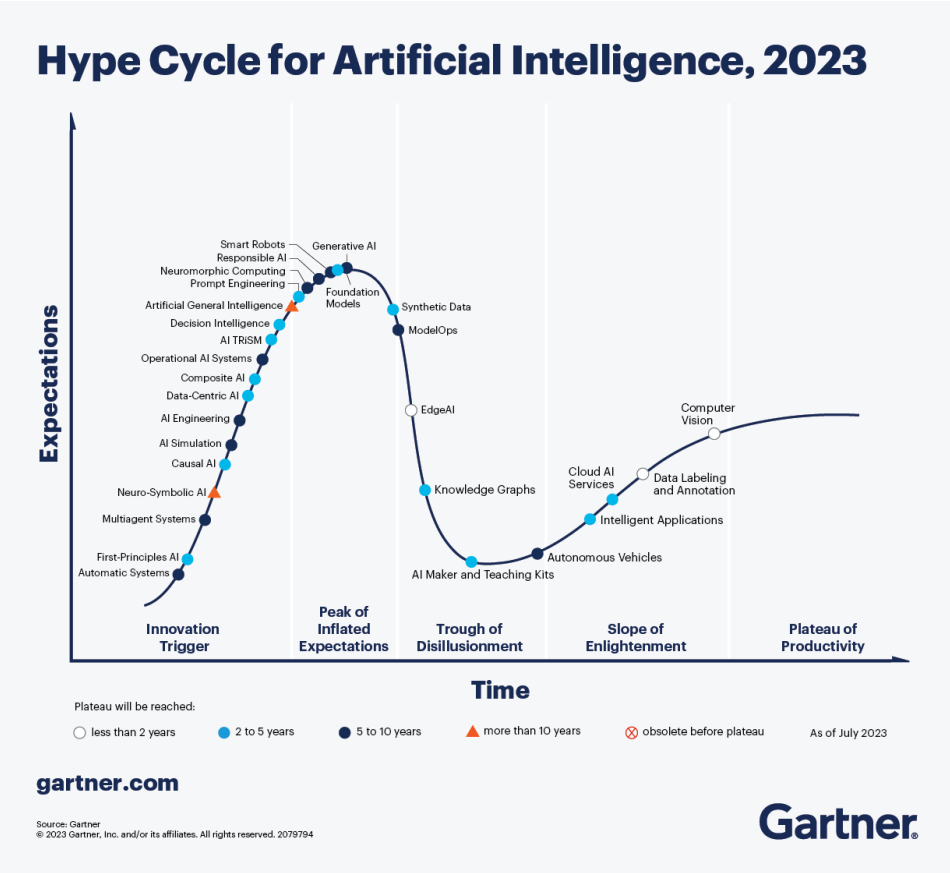

Figure 1 - Gartner Hype Cycle for AI, 2023 (Reference 1)

There are commentators with hypotheses ranging across the spectrum of possibility, from utopian solutions to predictions of Armageddon-like consequences. Let's consider the two most extreme scenarios in the world of pharmaceutical programming and follow industry knowledge to reach a likely conclusion.

Scenario 1: *It's all hype. Gen-AI will be abandoned and forgotten about in a few years' time.*
Scenario 2: *The computers are taking over; we will all be out of a job.*

Naturally the reality will lie somewhere between these most extreme of example. What are the possibilities, and restrictions that will drive (or inhibit) adoption?

INTRODUCTION TO GENERATIVE AI

The term "Generative AI" (Gen-AI) and brands such as Chat-GPT have been awash all forms of media in the last 18 months. Starting at the very beginning, Gen-AI is described as artificial intelligence capable of generating text, images or other media using generative models.

There are two key milestones in AI research that underpin Gen-AI:

LARGE FOUNDATION MODELS

A Large Foundation Model (LFM) can serve as a foundation for customization. These are expensive to create from scratch but once created they can be customised using relatively small additional data to achieve high-level performance without significant investment, allowing off-the-shelf foundations to create a Large Language Model.

LARGE LANGUAGE MODELS

A Large Language Model (LLM) is a type of Gen-AI created specifically to help generate text-based content, making them suitable for chatbots and programming co-pilots. A LLM is comparatively cheap and simple to train when building on an existing LFM, but this will depend significantly on the scope and scale of the language being trained.

DEMONSTRATION

The aim of this demonstration is to create a simple SAS macro program that can be repeatedly called to generate summary statistics for a specified variable and dataset. The following command resulted in the output in *Figure 2*.
"Create a SAS Macro program to calculate summary statistics".

The resulting output appears to the untrained eye is a suitable macro program. In the interest of balance, the following are *positive* observations of this code:

1. Balanced and consistent indentation.
2. Use of title case for the macro program name.
3. Defensive programming and error checking attempted.
4. Basic comments added throughout.

However, this first program created had some worrying issues, which if executed blindly **could cause data loss**:

1. The first data step overwrites the original source data.
2. The second step contains "if 0" meaning it will never execute the set statement.
3. VARTYPE function reference is incorrect (VARTYPE requires two numeric parameters).

```
%macro SummaryStats(data, var);
    /* Calculate summary statistics for a numeric variable */
    data &data;
        /* Ensure the variable exists and is numeric */
        if 0 then set &data;

        if vartype("&var") = 'N' then do;
            call symputx('var_found', 1);
        end;
        else do;
            call symputx('var_found', 0);
        end;
    run;

    /* Check if the variable exists and is numeric */
    %if &var_found = 0 %then %do;
        %put ERROR: Variable &var not found or is not numeric in dataset &data.;
    %end;
    %else %do;
        /* Calculate summary statistics */
        proc means data=&data n mean std min max;
            var &var;
        run;
    %end;
%mend;
```

Figure 2 - SAS Macro Program Generated by ChatGPT

These issues were easily corrected with subsequent commands. The most challenging part was to tell ChatGPT the alternative data step to be used for input validation.
Subsequent commands used were:

"Add a program header".
This created the program header exactly as shown below. The description is detailed enough and provides an example of the macro's usage.

"Replace the first data step with this code…".
I specifically provided the correct data step to replace the erroneous error checking dataset. ChatGPT correctly identified and updated the program. However, it removed the program header created from the previous request.

"Add a program header to the previous macro".
This added the program header to the previous response.

"Add a data step to this macro that checks if the dataset exists".
ChatGPT performed as expected and added an additional %IF-%THEN statement, rather than a data step. The ready program code is shown below (*Figure 3*):

```
/*****************************************
* SAS Macro Program for Summary Statistics *
*****************************************
This macro calculates summary statistics for a numeric variable in a given dataset.
It calculates the mean, standard deviation, minimum, and maximum values for the
specified variable and prints the results in the SAS log.

Usage:
- Specify the name of the dataset as 'data'.
- Specify the name of the numeric variable as 'var'.

Example Usage:
%SummaryStats(data=your_dataset, var=your_variable);
*/

%macro SummaryStats(data, var);
    /* Check if the dataset exists */
    %if %sysfunc(exist(&data)) = 0 %then %do;
        %put ERROR: Dataset &data does not exist;
        %return;
    %end;

    /* Calculate summary statistics for a numeric variable */
    data _null_;
        dsid = open("&data.");
        n = varnum(dsid, "&var.");
        x = vartype(dsid, n);
        rc = close(dsid);

        if x = 'N' then do;
            call symputx('var_found', 1);
        end;
        else do;
            call symputx('var_found', 0);
        end;
    run;
```

```
    /* Check if the variable exists and is numeric */
    %if &var_found = 0 %then %do;
        %put ERROR: Variable &var not found or is not numeric in dataset &data.;
    %end;
    %else %do;
        /* Calculate summary statistics */
        proc means data=&data n mean std min max;
            var &var;
        run;
    %end;
%mend;

/* Example usage of the macro */
%SummaryStats(data=sashelp.class, var=Weight);
```

Figure 3 - Corrected SAS Macro Program

Whilst this is a simple example, it demonstrates both the power and risks of using Gen-AI to write code. This program was created with a handful of single sentence commands but without the correct knowledge and oversight, had the potential to introduce a significant error.

OPPORTUNITY, RESPONSIBILITY AND ACCOUNTABILITY

Setting aside the hype of Gen-AI, the main challenges of adopting any use of new technology are of accepting change and managing risk within your company, just as has been experienced in many instances over the years. Consider the following stakeholders for change in pharmaceutical programming:

- Programmers
- Quality Assurance managers
- Industry regulators
- Clients / customers

Each of the stakeholders will have their own sight of trends and changes in technology and without necessarily being a blocker, it is essential to have all stakeholders involved and successful use of AI.

PROGRAMMERS

The demonstration in the previous section demonstrates how easily a programmer can create a new SAS program, or the outline of one at very least. Programmers of all languages will be well used to searching online documentation, be this from a software vendor's official documentation, or an online community. There is new opportunity for programmers to seek answers to problems much more quickly without having to scroll through various forum pages or internet search results to find a relevant issue.

The limitation is that Gen-AI tools such as ChatGPT is trained on publicly available information such as those forums and documentation sites users commonly search. As such, there is no guarantee that Gen-AI will provide better answers and will not have more knowledge than is available already. The benefit should come from Gen-AI being better able to provide the most appropriate response immediately, although this is far from a given. Below is an example of a requesting for help from ChatGPT that could also be sought from SAS Documentation:

Question:

"Provide an example of the VARTYPE function in SAS".

Response:

"As of my last knowledge update in September 2021, there is no built-in VARTYPE function in SAS. If you are looking for a way to determine the data type of a variable in SAS, you typically don't have a direct function for this purpose".

ChatGPT then provided the following code sample to achieve what it thinks I was looking for.

```
data mydata;
  input myvar $;
  datalines;
123
ABC
;
run;
```

```
data check_var_type;
  set mydata;
  if not missing(myvar) then do;
    /* Attempt to convert to numeric, if successful, it's numeric */
    myvar_num = input(myvar, ?? best32.);
    if not missing(myvar_num) then var_type = 'Numeric';
    else var_type = 'Character';
  end;
run;

proc print data=check_var_type;
run;
```

Firstly, SAS **does** have a function named VARTYPE (reference 2). Further, this response will perform a summary of values being numeric or character from which we could infer a numeric/character column, but would be an inefficient method for achieving it, especially on large datasets.

This demonstration provides another example the limitations of open tools like ChatGPT. No guarantee is offered for the accuracy of answers given to questions. We've already seen syntax errors and dangerous code in the examples above.

There are already AI-driven tools available to almost every programmer. Co-pilot tools in products such as GitHub can already aid with generating code. Further, SAS announced at the recent SAS Explore conference (reference 3) that their own copilot is in development. Will a co-pilot tool generated and provided by a specific software vendor like SAS offer any improvements in guarantees of accuracy, arguably not if the training data is no clearer than what is publicly available.

The programmer must maintain responsibility for any code being produced and thus be accountable for any program's performance and function.

QUALITY ASSURANCE

CRO's and Pharmaceutical programmers use independent parallel programming as the de-facto standard method of quality assurance for any clinical datasets (reference 4). QA managers are responsible for ensuring strict operating procedures and work instructions are in place and programmers are trained accordingly to ensure analysis programs are written independently and without bias or influence from one programmer to another. There are arguments for and against the quality assurance impact of Gen-AI code:

For: This is not really any different to programmers Googling or searching for code help online, as ChatGPT is trained on internet data.

Against: Coding styles risk becoming much more homogenous and risk undermining *true* independence between programmers.

There is merit in both the above statements. However most (if not all) pharmaceutical companies and CROs use a suite of standardised code libraries, be this written in SAS, R etc for streamlining the common analytical programming jobs. These usually comprise many macro programs such as the demonstration above. As such, a question arises as to whether using Gen-AI tools for programming is even relevant. Such tools will create textbook code based on public documentation but will have no knowledge of any company's internal code libraries.

This opens the door to creating and training a custom LLM based on a company's existing coding standards and libraries thus creating a form of Gen-AI that is bespoke to support the needs of your company's programmers. Any such custom model could be trained and validated to keep control of quality. The financial viability of training a custom LLM very much separate debate.

REGULATORS

Regulatory requirements impact large amounts of day-to-day work in the pharmaceutical sector. The FDA and EMA are actively assessing the impact of AI. In 2021, the FDA reported that there were over 100 submissions reported that used some form of AI in drug the development (from drug discovery, clinical research, post-market safety surveillance and manufacturing). In May 2023, FDA further published a discussion paper on Artificial Intelligence and Machine Learning for Drug Development (reference 5) and request for feedback. However, there is minimal focus on the use of Generative AI, instead the focus is on use of AI alongside Digital Health Technologies. This opportunity for feedback provides scope for the pharmaceutical programming industry to raise awareness of opportunity and risks associated with regulators.

Lessons from History

The advent of Gen-AI shares some comparisons with previous technologies that have emerged in past years that we now take for granted. An internet search will quickly reveal articles from journalists predicting impact to international

employment figures with various statistics being quoted (references 6, 7). But are these claims substantiated? Similar articles and claims have been made since advent of modern computers since the 1970s, but overall reality is one of creating opportunities for improved productivity over mass unemployment. A report by PwC for the UK government in 2021 (reference 8) concludes minimal impact in the overall employment but improvements in productivity and real income are possible. Of course, this report covers far wider areas of AI than pharmaceutical programming.
A lesson from the SAS world is to look back 20 years to the release of SAS Enterprise Guide. While conference papers at the time (references 9, 10) focussed on the improved productivity of code automation through a point-and-click interface, some feared the days of the programmer were numbered. Comparisons can be made around a reduced need for in-depth knowledge of the SAS language. Unfortunately, as a private company SAS don't publish numbers estimating the number of customers and users globally or by sector. However, we can see that the first PhUSE conference in 2005 had a mere 270 attendees and has since doubled to over 600 in 2022.

CONCLUSIONS

Generative AI tools are very powerful for generating code automatically. It is critical though that any automation of code is carefully controlled and overseen by quality processes and individuals with relevant technical expertise. Gen-AI can create large amounts of content very quickly from minimal instructions but does not yet have the skill or depth of knowledge of an experienced programmer.

The use of AI in the wider pharmaceutical sector has had more focus on increasing speed of the overall drug development process, with examples of identifying AE's, interpreting scans rather than a Gen-AI focus. It is incumbent on those in pharmaceutical programming to ensure industry regulators are aware of the possibilities while such opportunity for feedback is open.

Wide-scale use of Gen-AI and Large Language Models is relatively in its infancy. Further maturing of these technologies over time is inevitable including improvements in AI coding. As an evidence-based industry there will be a requirement for clear evidence of Gen-AI achieving parity at least with humans before the days of the programmer are numbered. In the meantime, Gen-AI tools such as ChatGPT and coding co-pilots have the potential to serve as an accelerator to all pharmaceutical programmers, whether using SAS, R, Python, or any other programming language.

Red Pill or Blue Pill? Gen-AI exists and cannot be un-invented. Technology will mature over time and become more and more prevalent to the pharmaceutical programmer in their day-to-day role. While it may not replace the programmer for a very long time, the industry has already taken red pill.

REFERENCES

1. What's New in Artificial Intelligence From the 2023 Gartner Hype Cycle™
2. SAS Help Center: VARTYPE Function
3. About - SAS Explore 2023
4. 83_Final_Paper_PDF.pdf (lexjansen.com)
5. Artificial Intelligence and Machine Learning (AI/ML) for Drug Development | FDA
6. Rise Of Robots - Jobs Lost to Automation Statistics in 2023 (leftronic.com)
7. Artificial Intelligence Has Caused A 50% To 70% Decrease In Wages—Creating Income Inequality And Threatening Millions Of Jobs (forbes.com)
8. The impact of AI on jobs (publishing.service.gov.uk)
9. SUGI 25: Enterprise Guide Development: Enterprise Client Software for the Windows2 Platform (sas.com)
10. SUGI 26: Who Needs to Know Program Syntax When You've Learned Enterprise Guide (sas.com)

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Paul Shannon
Katalyze Data Ltd.
11 Wesley Walk
Witney
Oxfordshire, OX28 6ZJ
Work Phone: +44 (0) 1993 848010
Email: paul.shannon@katalyzedata.com
Web: https://katalyzedata.com