

# SAS Programmers Need AI

Craig Parry  
MAC Clinical Research  
craigparry@macplc.com

## 1 INTRODUCTION

In November 2022, the first 'mainstream' AI chatbot, Chat GPT was launched and looking at Google trends, the term 'AI' saw a noticeable spike around this time. We then saw a larger spike in April 2023 due to the release of the next version of Chat GPT, version 4.0.

The trend is similar to the '3D' movie fad that has appeared multiple times, but 'AI' seems to carry more value and longevity as opposed to being seen as a gimmick, and we are already seeing companies begin to integrate AI into their systems. Examples include AI being used to successfully process MRI images<sup>(3)</sup> and to assist with medical coding<sup>(4)</sup>.

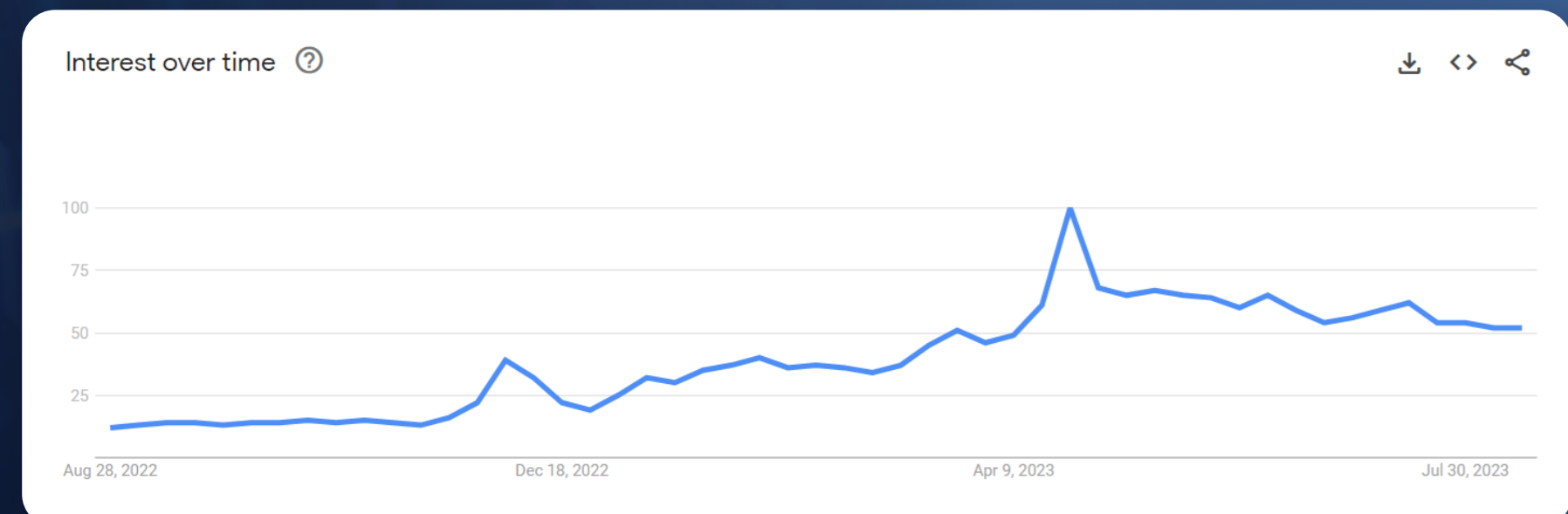


Figure 1: Google trends of the term AI over the last 12 months.  
Sources:  
(3) <https://www.ox.ac.uk/news/features/how-artificial-intelligence-shaping-medical-imaging>  
(4) <https://medcitynews.com/2023/06/what-every-health-system-cfo-needs-to-know-before-using-ai-for-medical-coding/#:~:text=Fully%20autonomous%20coding%20means%20that,you%20pay%20across%20the%20board>



## 2 FUNCTION VS. FORM

It is quite common to be assigned a study that requires a little bit of a 'rescue' - however 'ample time' and 'rescue' rarely appear in the same sentence. This means that we often have little time to digest the study and attempt to understand the pre-existing programs. This is where AI can step in and assist as it is a resource that doesn't get sick, stressed, or exhausted. Figure 2 highlights how AI can quickly untangle complex code.

Figure 2: AI correcting code to ensure good programming practices are used.

Code Written by a "Programmer"
<pre>proc sql noprint; create table cars as select * from sasHELP.cars order by make, type; create table cars_2 as select a.make, b.type, a.max_engineize, b.total_engineize from (select make, max(engineize) as max_engineize from cars group by make) as a left join (select make, type, min(engineize) as total_engineize from cars group by make, type) as b on a.make=b.make; quit;</pre>
Question Asked to AI
<p>Could you reformat this code to follow good programming practices without adding anything to the code?</p>
Code Corrected by AI
<pre>proc sql noprint; create table cars as select * from sasHELP.cars order by make, type; create table cars_2 as select a.make, b.type, a.max_engineize, b.total_engineize from (select make, max(engineize) as max_engineize from cars group by make ) as a left join (select make, type, min(engineize) as total_engineize from cars group by make, type ) as b on a.make = b.make; quit;</pre>

## 3 HYPERREALISM VS. SURREALISM

In figure 2 AI has successfully corrected a poorly written piece of code although it still resembles something similar to Salvador Dali's artwork. You can see a clock but struggle to comprehend why it's melting. Our programs should be universally understandable which is why we incorporate comments. However, some programmers believe that no one will ever revisit their code so there's no need for comments. That's a misconception as leaving comments is essential in all instances and it's virtually impossible to have

too many comments. So, we encounter our second challenge for AI to address: dealing with the absence of comments in code.

Question Asked to AI
<p>Could you add some comments to the code to help me understand?</p>
Code Corrected by AI
<pre>proc sql noprint; /* Create a new table 'cars' from the database 'sasHELP.cars'. This table sorts the cars based on their make and type. */ create table cars as ...; /* Create a new table 'cars_2' that will contain: - The make of the car - The type of the car - The maximum engine size for each make - The total engine size for each make and type combination */ create table cars_2 as ...; /* This subquery calculates the maximum engine size for each make. The result is aliased as 'a' for reference in the main query. */ (select make, max(engineize) as max_engineize... /* This subquery calculates the total engine size for each make and type combination. The result is aliased as 'b' for reference in the main query. */ (select make, type, min(engineize) as total_engineize... quit;</pre>

Figure 3: AI successfully adding comments to help us understand the code. Some code has been removed for ease of reading.

## 4 COMPLICATED VS. SIMPLE

In figure 3 the code is well-structured, with helpful comments, and it's easy to comprehend. However, what remains unclear is why someone opted to overcomplicate such a basic problem. I often come across intricate PROC SQL code, especially nested PROC SQL, only to realise that this seemingly 'advanced' piece of code accomplishes a fundamental task in just a few lines. Lengthy programs don't necessarily imply poor quality; the crucial point is that the code should be universally understandable.

AI can certainly streamline and help make sense of messy code but is this truly beneficial? Or is AI enabling programmers to adopt a 'quick fix' mentality? For AI to be genuinely effective, it should focus not just on doing the job but also on educating the user.

Question Asked to AI
<p>Could you do the same task this code is doing without using PROC SQL?</p>
Code Corrected by AI
<pre>/* Step 1: Sort the sasHELP.cars dataset by make and type */ proc sort data=sasHELP.cars out=cars; by make type; run; /* Step 2: Calculate max_engineize for each make */ proc means data=cars noprint max; var engineize; by make; output out=max_engineize max(engineize)=max_engineize; run; /* Step 3: Calculate total_engineize for each combination of make and type */ proc means data=cars noprint sum; var engineize; by make type; output out=total_engineize sum(engineize)=total_engineize; run; /* Step 4: Merge datasets together based on make */ data cars_2; merge max_engineize (now) total_engineize (now); by make; /* Only keep rows where there's a match from both datasets */ run; quit;</pre>

Figure 4: AI simplifying the overly complicated, much better.

## 5 RATIONALE THOUGHTS PLEASE

It is important to note that in every example above (figures 2, 3 and 4), I did not include the rationale that the AI autogenerated. This rationale is the AI both defending and explaining its changes. So, let's revisit figure 2 where we asked the AI to adjust the original code to follow good programming practices and examine the rationale provided.

In figure 2B this is a great rationale as it is educating me on why the changes were made and the importance of consistency and clarity within a team.

The code the AI generated included comments prefixed with step 1, step 2, step 3, and step 4. This is because the rationale was written as steps to help the user understand each change made.

Question Asked to AI
<p>Could you reformat this code to follow good programming practices without adding anything to the code?</p>
Rationale for Changes Provided by AI
<p>Here's a summary of what I've done:</p> <ol style="list-style-type: none"> <li><b>Comments:</b> Provided comments to give context to what each step or subquery does.</li> <li><b>Indentation:</b> Used consistent indentation for better readability.</li> <li><b>Spacing:</b> Provided appropriate spacing between different parts of the SQL statement.</li> <li><b>Alignment:</b> Aligned select, from, and group by clauses for better readability of subqueries.</li> </ol> <p>Remember that good programming practices often vary by team or organization, so always consider the standards in your specific environment.</p>

Figure 2B: Rationale provided by AI omitted from figure 2, which explains why the AI did what it did when improving the original codes layout.

## 6 SPOT THE DIFFERENCE

However, this overlooks the possible issue of a language barrier. Rationale alone is not the best way for AI to explain the differences between the original code and the AI-corrected code, because rationale assumes everyone's understanding of the English language is the same.

SAS® is a programming language and like any language we see differences. For example, my American counterpart and I both speak English but there are differences, e.g., Haematology vs. Hematology. These subtle differences are also seen in programming and AI will have its own preference when writing code. Could AI visually highlight the differences without using comments or rationale?

Let's take the original program and ask the AI to do something else. We still want to improve the code, but we also want to highlight the changes in a universally readable method, ideally without writing out a comment. A visual and clear indication of what the AI has changed.

Asking the AI "You know the code I originally showed you, and then the improved code you showed me, are you able to visually highlight the differences?" returns an interesting result. Note, the original code I am referring to is the code seen in Figure 2. Asking the AI this returns an interesting result, AI asking me a question "Would you like a side-by-side comparison in this style for the entire code?". Yes, I 100% would!

Which leads us to figure 6. Now, there is an obvious issue with the comparison as the original code (seen in figure 2) went from 11 lines of code

to 20 lines, but the visual comparison mentions the text [New Line] only once. Then again it could be argued the AI did "visually show us the differences" and paired with the rationale, I think the AI did an OK job.

Question Asked to AI
<p>Yes please, but could you also highlight any new lines introduced?</p>
Answer Provided by AI
<pre>proc sql noprint; [Indent] create table cars as [Indent] select * [Indent] from sasHELP.cars [Indent] order by make, type; [New Line] [Indent] create table cars_2 as [Indent] select a.make, b.type, a.max_engineize, b.total_engineize [Indent] from [Indent] (select make, max(engineize) as max_engineize [Indent] from cars [Indent] group by make [Indent] ) as a [Indent] left join [Indent] (select make, type, min(engineize) as total_engineize [Indent] from cars [Indent] group by make, type [Indent] ) as b [Indent] on a.make = b.make; quit;</pre>

Figure 6: Asking AI to show me its visual "spot the difference" attempt of the changes seen in figure 2 between the original code and the AI-improved code.

## 7 FEED IT ERRORS

Programmers experience programming coding issues every day so let's present the AI with a very common issue: date conversion.

I want to convert my character date variable, of the format DDDMMYYYY (e.g., 11AUG2023) into a numeric variable but my code is not working. Let's see if the AI can help.

In figure 7 this is correct and the rationale explains the reason why the error may have appeared. But I did have to provide a very precise question, which is not ideal.

Question Asked to AI
<p>Slightly off topic, I am trying to convert a character date variable in SAS of the format DDDMMYYYY, to be numeric, using the following code:</p> <pre>date_num = put(date_c, yymmdd10.);</pre> <p>However, I am getting the following note in the log and the variable in my output dataset is not numeric:</p> <pre>NOTE: 484-185: Format SYMMDD01 was not found or could not be loaded.</pre> <p>Can you help me?</p>
Answer Provided by AI
<p>Certainly, the issue you're facing is due to the use of the <code>put</code> function, which is trying to apply a numeric date format (<code>SYMMDD01</code>) to a character variable. To make <code>date_num</code> be interpreted as a date, you're looking for a character date format, hence the <code>SYMMDD01</code> in the log note. This is not what you want.</p> <p>If you're trying to convert a character date variable of the format <code>DDMMYYYY</code> to a numeric date variable, you should be using the <code>input</code> function instead of <code>put</code>.</p> <p>Here's the correct code:</p> <pre>date_num = input(date_c, date9.);</pre> <p>The <code>date9.</code> informat reads the date in <code>DDMMYYYY</code> format and converts it to a numeric date variable that represents the number of days since January 1, 1960.</p>

Figure 7: AI successfully providing a solution to a date conversion issue.

## 8 PRECISION IS KEY

You must be precise when bringing your problem to an AI. Take the date conversion prompt seen in section 7 in figure 7 as the question posed is rather detailed:

- The input date format is mentioned i.e., DDDMMYYYY.
- SAS® is explicitly mentioned.
- We have also defined the input variable as character.
- I also show the AI code I am attempting to use.

Without providing the AI with this level of detail, it will typically get things wrong as seen in Figure 8. I should be clear that the AI is not wrong, what it is saying is correct.

However, my input date variable is of the format DDDMMYYYY so using the format YYMMDD10 in the INPUT function would not work in my code. The fault is with our question not the AI.

Additionally, AI often likes to provide additional information without being asked to do so (figure 7). I never asked about adding a format, but it told me about the FORMAT statement anyway.

In contrast the AI would forget snippets of earlier questions and a good example of this is my first question in figure 2 clearly mentioning SAS® programming. By the time I got to my question in figure 7 the AI ignored this fact and began showing me answers in different programming languages.

One of the key benefits of AI, that it can think for itself, can sometimes present unwanted results.

Question Asked to AI
<p>Slightly off topic, I am trying to convert a date variable however, I am getting the following note in the log and the variable in my output dataset is not numeric:</p> <pre>NOTE: 484-185: Format SYMMDD01 was not found or could not be loaded.</pre> <p>Can you help me?</p>
Answer Provided by AI
<ol style="list-style-type: none"> <li>1. <b>Correcting a Character Date to a Numeric Date:</b> If your date is stored as a character string and you want to convert it to a SAS date (numeric), you'd use the <code>INPUT</code> function.</li> <li>2. <b>Applying a Format to Display a Numeric Date:</b> Once you have a numeric date, you can use the <code>FORMAT</code> statement to control how it's displayed.</li> </ol> <pre>data want; set have; /* Convert character date to numeric date */ date_num = INPUT(date_char, yymmdd10.); /* Apply format to the numeric date */ FORMAT date_num SYMMDD01.; run;</pre>

Figure 8: AI providing an attempt at a solution when I pose a vague question.

## 9 CONCLUSION

The straightforward conclusion is that AI is more accessible than ever before. If you are concerned about AI taking your job, then I would suggest that you might be in the wrong profession and AI should not be your primary worry. I can confidently state that AI has the potential to train and educate the next generation of programmers while assisting the current cohort. It can help overcome language barriers and alleviate some of the stress associated with problem-solving.

I recently conducted a brief survey on LinkedIn posing a simple question: 'How will Artificial Intelligence (AI) impact your job?' The available answers were 'Positively' or 'Negatively.'



The consensus suggests that AI is more of an asset to the majority rather than an 'inevitable replacement'.

Scan QR code to read full paper