

---

# HOW GENERATIVE AI MODELS SUCH AS CHATGPT CAN BE (MIS)USED IN SPC PRACTICE, EDUCATION, AND RESEARCH? AN EXPLORATORY STUDY

---

A PREPRINT

Fadel M. Megahed<sup>1</sup>, Ying-Ju Chen<sup>2</sup>, Joshua A. Ferris<sup>1</sup>, Sven Knoth<sup>3</sup>, and L. Allison Jones-Farmer<sup>1,\*</sup>

<sup>1</sup>Farmer School of Business, Miami University, Oxford, OH 45056, USA

<sup>2</sup>Department of Mathematics, University of Dayton, OH 45469, USA

<sup>3</sup>Department of Mathematics & Statistics, Helmut Schmidt University, Hamburg, Germany

\*Corresponding author. Can be reached at [farmerl2@miamioh.edu](mailto:farmerl2@miamioh.edu)

February 16, 2023

## ABSTRACT

Generative Artificial Intelligence (AI) models such as OpenAI's ChatGPT have the potential to revolutionize Statistical Process Control (SPC) practice, learning, and research. However, these tools are in the early stages of development and can be easily misused or misunderstood. In this paper, we give an overview of the development of Generative AI. Specifically, we explore ChatGPT's ability to provide code, explain basic concepts, and create knowledge related to SPC practice, learning, and research. By investigating responses to structured prompts, we highlight the benefits and limitations of the results. Our study indicates that the current version of ChatGPT performs well for structured tasks, such as translating code from one language to another and explaining well-known concepts but struggles with more nuanced tasks, such as explaining less widely known terms and creating code from scratch. We find that using new AI tools may help practitioners, educators, and researchers to be more efficient and productive. However, in their current stages of development, some results are misleading and wrong. Overall, the use of generative AI models in SPC must be properly validated and used in conjunction with other methods to ensure accurate results.

*Key Words:* artificial intelligence; innovation engineer; large-language models; prompt engineering

## 1 Preface

“What do you think of ChatGPT?” Many readers of this paper have asked and/or been asked this question, likely sometime between December 2022 and January 2023. Writing an article about large language models (LLM), such as ChatGPT, is a daunting task since: (a) the technology is rapidly evolving; (b) the content can cover many aspects, including the details of a given model and ethical/legal considerations about the role of artificial intelligence (AI) in our society; and (c) it is hard to forecast the ubiquitousness of such tools in a few years. In this paper, we focus on what LLM models can and cannot do well now.

The overarching objective of this expository paper is to examine whether LLMs such as ChatGPT-like models **can be useful in statistical process control (SPC) applications**. Our examination will focus on three primary SPC application domains: **practice, learning/training, and research**. We do not discuss important issues such as ethical, legal, and other philosophical concerns from using AI tools.

## 2 Introduction

The rapid developments in AI over the past three years have changed the general perception of what AI can do. In the SPC community, we have equated AI to predictive machine learning (ML) systems that are trained to solve classification, regression, or clustering problems (Colosimo et al., 2021; Megahed and Jones-Farmer, 2015; Weese et al., 2016). While ML remains an essential component of AI, state-of-the-art models have transitioned to “generative AI” where the objective is to generate new content rather than analyze an existing dataset (Gozalo-Brizuela and Garrido-Merchan, 2023). The generated content is based on a stochastic behavior embedded in generative AI models such that the same input prompts results in different content.

State-of-the-art generative AI models can have up to 175 billion parameters. With the increase in model size, researchers have observed the “emergent abilities” of LLMs, which were not explicitly encoded in the training (Wei et al., 2022). Examples of “prompted tasks with emergent performance [included]: multi-step arithmetic, taking college-level exams and identifying the intended meaning of a word.” (Wei and Tay, 2022)

These models have allowed a wide range of applications. For example, GPT-3 has been used in language translation, text summarization, and content generation applications (Devlin et al., 2018). To reduce the level of expertise needed to use and deploy those generative AI models, AI-powered chatbots and virtual assistants, such as ChatGPT and CoPilot, have been developed (McKee and Noever, 2022). For example, ChatGPT is a chatbot-like variation of GPT-3 (technically GPT-3.5) that has been fine-tuned for conversational language understanding and generation, allowing it to respond more human-like in a conversational context. In addition to text-to-text generative AI tools, several breakthroughs in text-to-audio (e.g., *Google’s MusicLM* (Agostinelli et al., 2023)), text-to-image (e.g., *OpenAI’s DALL.E 2* (OpenAI, 2022c)), and text-to-video (e.g., *Meta’s Make-A-Video* (Singer et al., 2022) and *Make-A-Video3D* (Singer et al., 2023)) applications have been made.

The recent generative AI breakthroughs (see Figure 1a) and adoption of these tools (as in ChatGPT’s case depicted in Figure 1b) have created hype in the business community. *Sequoia Capital* estimates that “generative AI can make [knowledge and creative] workers at least 10% more efficient and/or creative: they become not only faster and more efficient, but more capable than before. Therefore, Generative AI has the potential to generate trillions of dollars of economic value.” (Huang et al., 2022) Their viewpoint reflects the broader investor community’s sentiment as it invested \$1.37 billion in generative AI in 2022, approximately equal to their combined investment in the previous five years (Griffith and Metz, 2023).

In this article, we ask, “what can generative LLM-based AI tools do now to augment the roles of SPC practitioners, educators, and researchers?” To make our task more tractable, we will primarily focus on evaluating the utility of ChatGPT (and its underlying GPT-3.5 engine) since it: (a) is the most well-known of these generative AI tools and (b) combines features of the generative chatbot with an underlying LLM that can generate both text and code. In our estimation, this expository assessment can provide a benchmark for future evaluations of the next generation of generative AI models. We hope this study motivates SPC practitioners and researchers to be receptive to exploring whether new AI tools can help them be more efficient, productive, and innovative. This is consistent with the recommendations of (a) Box and Woodall (2012, p. 20) who stated that “we stress the necessity for the quality engineering community to strengthen and promote its role in innovation”, and (b) Hockman and Jensen (2016, p. 165) who stated that “for statisticians to be successful in leading innovation, they will need to strengthen their skills beyond what they have traditionally needed in the past, but we believe this will be worth the effort”.

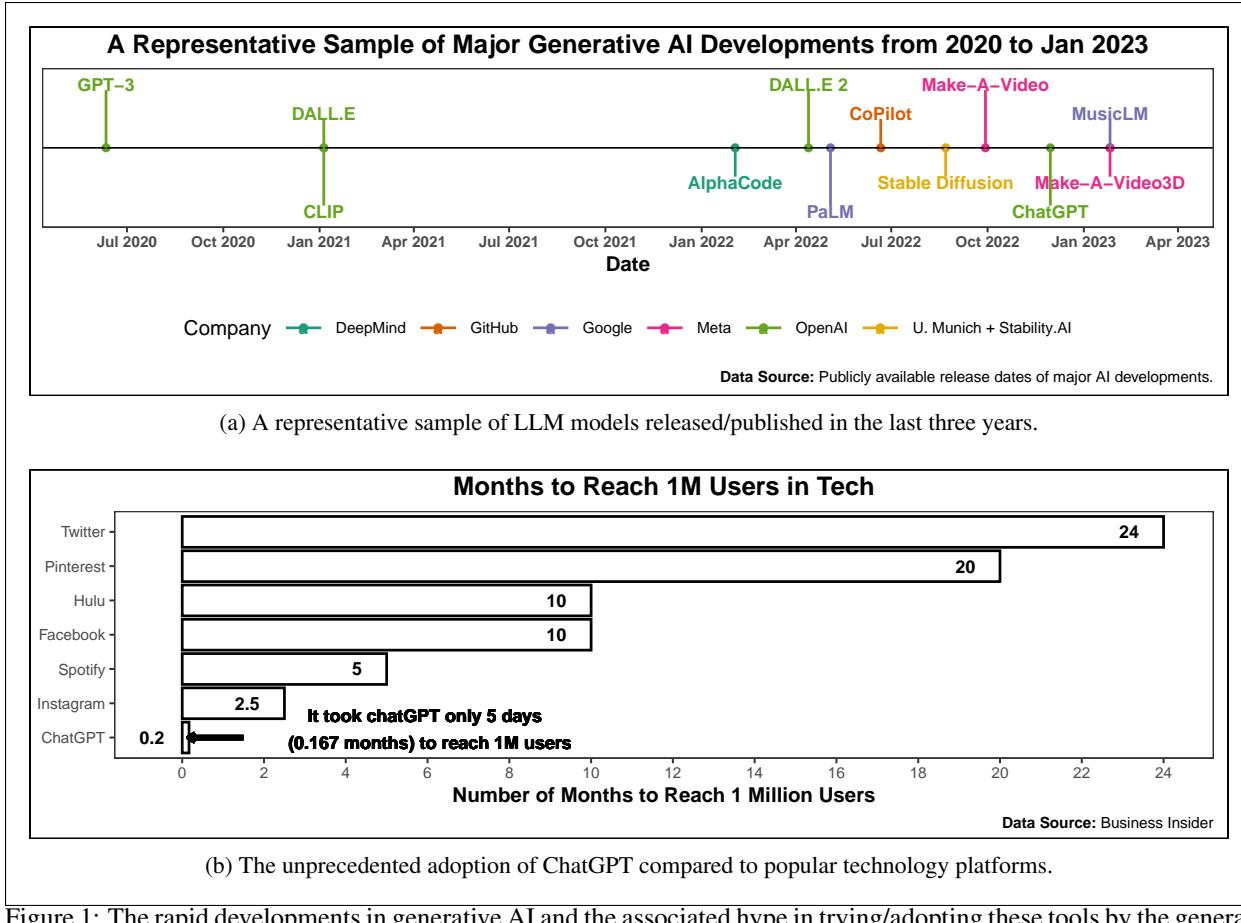


Figure 1: The rapid developments in generative AI and the associated hype in trying/adopting these tools by the general public.

### 3 Background

The notion of a language model has risen in popularity in the last decade thanks to modeling techniques and computational power advances. Language models are used to assign probabilities to word sequences. Interestingly, the foundation of language models was set by the mathematician Andrey A. Markov in the early 1900s, who used Alexander Pushkin's Eugene Onegin novel to demonstrate that letter pair sequences are not independent and that the likelihood of a letter's appearance can be approximated (Markov, 2006). Shannon (1948) built on that work, developing a statistical model for the English language and showing that text sequences could be generated. The work of Markov and Shannon is the foundation of modern language work; modern models will use sequences of words rather than events (Markov) or characters/words (Shannon). Today, language models are ubiquitous. Examples include text auto-completion (used in web browsers, mobile phones' messaging apps, software GUIs, etc.), machine translation, natural language generation, and optical character recognition.

The development of the *Transformer* model set the foundation for today's large language models. The *Transformer* model was proposed in Vaswani et al. (2017) as a deep learning model grounded on self-attention mechanisms. The self-attention mechanism assigns different weights according to the importance of each part of the data. The *Transformer* model architecture allows for parallel computing, thus reducing training time (Vaswani et al., 2017). The parallelization advantage and the model's architecture result in better performance and accuracy than traditional recurrent neural network-based deep learning models. Hence, existing LLMs utilize a *Transformer*-based architecture, making it possible to train multi-billion parameter models on several terabytes of text.

OpenAI, an AI research lab, introduced its third-generation autoregressive language model, *Generative Pre-trained Transformer 3* (GPT-3), which employs deep learning to generate human-like text in 2020 (Brown et al., 2020). GPT-3 was pre-trained on a diverse corpus of unlabeled contiguous text containing almost 500 billion tokens, and the model had 175 billion parameters making it, at the time, the largest natural language processing (NLP) model (Radford

et al., 2018). GPT-3 learns patterns and relationships in the data without a label, meaning the model will have no understanding of fact or fiction, just the patterns of words. During training, it predicts the next word in a text string, given the previous words as input. The model's parameters are optimized using the gradient descent algorithm, which improves its ability to generate text that resembles the text in the training data over time. Once trained, the model can be refined by adding task-specific data to train the model for specific NLP tasks such as text generation, summarization, translation, and question answering. By combining pre-training on a large corpus of diverse text data and fine-tuning on task-specific data, GPT-3 can perform various NLP tasks with high fidelity.

GPT-3 has proven to be a powerful tool for natural language, Brown et al. (2020) noted in their paper the potential to apply the model to programming applications. OpenAI created the Codex model, a fine-tuned GPT-3 for programming tasks with 12 billion parameters, and trained using select files from 54 million public GitHub repositories (Chen et al., 2021). Codex powers GitHub Copilot, a cloud-based tool embedded into several Integrated Development Environments (IDEs) such as Visual Studio, Visual Studio Code, JetBrains, and Neovim.

ChatGPT is a pre-trained instance of GPT-3, specifically optimized for the task of generating conversational responses. While GPT-3 can be used for various NLP tasks, ChatGPT is designed to generate human-like responses to the text input conversationally. Over time, the model learns the patterns and relationships in the data, improving its ability to generate text that resembles the text in the training data. This mechanism allows ChatGPT to generate consistent, contextually relevant responses to a wide range of natural language input. Figure 2 shows the ChatGPT training process (OpenAI, 2022a).

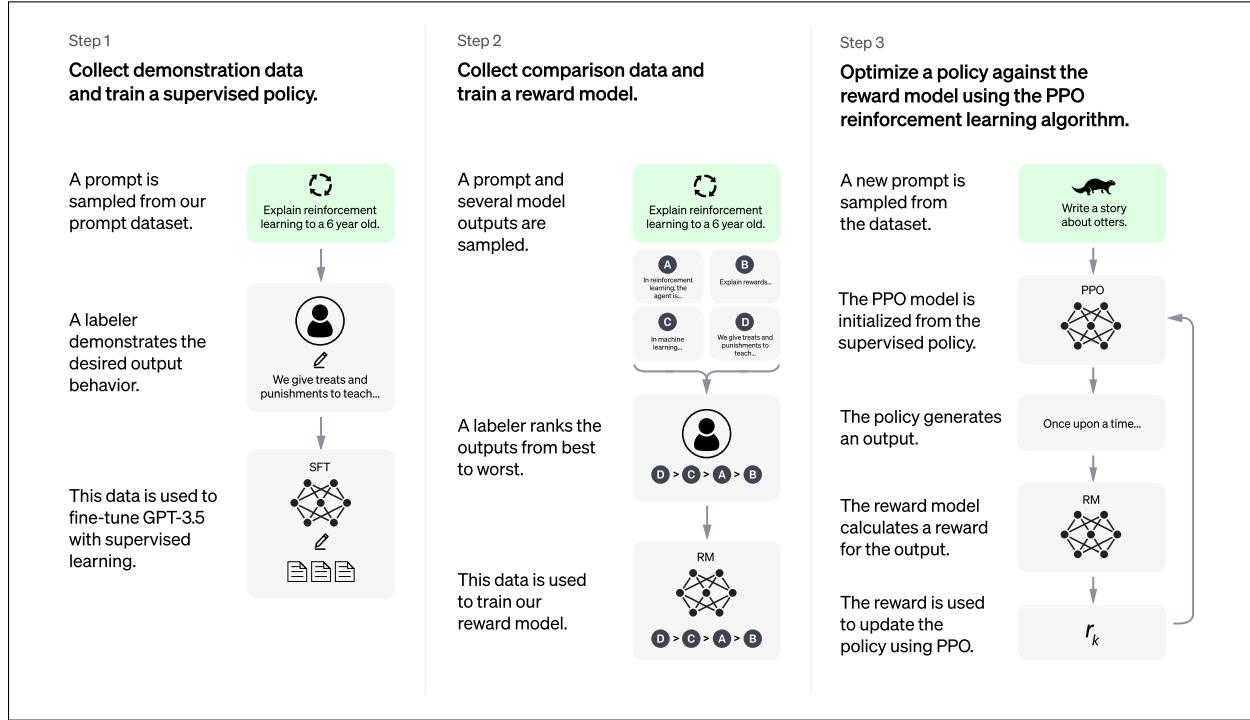


Figure 2: The ChatGPT training process. The figure is from OpenAI (2022a).

## 4 Designing our Exploratory Study

For this study, we focus on using SPC in three areas relevant to SPC: practice, learning, and research. We evaluate ChatGPT's ability (1) to provide code for a specified task; (2) to explain basic concepts; and (3) to create knowledge related to each of the three areas.

To demonstrate ChatGPT's ability to generate code, we evaluated several prompts related to creating control charts for practice, explaining existing code, translating code between languages for learning, and developing simulation code for research. To investigate ChatGPT's ability to provide correct and meaningful explanations of key concepts, we evaluated prompts related to defining and explaining SPC terms in practice, learning, and research. Finally, to assess the ChatGPT's ability to create new knowledge assets, we prompted it to create a framework for SPC, a syllabus for

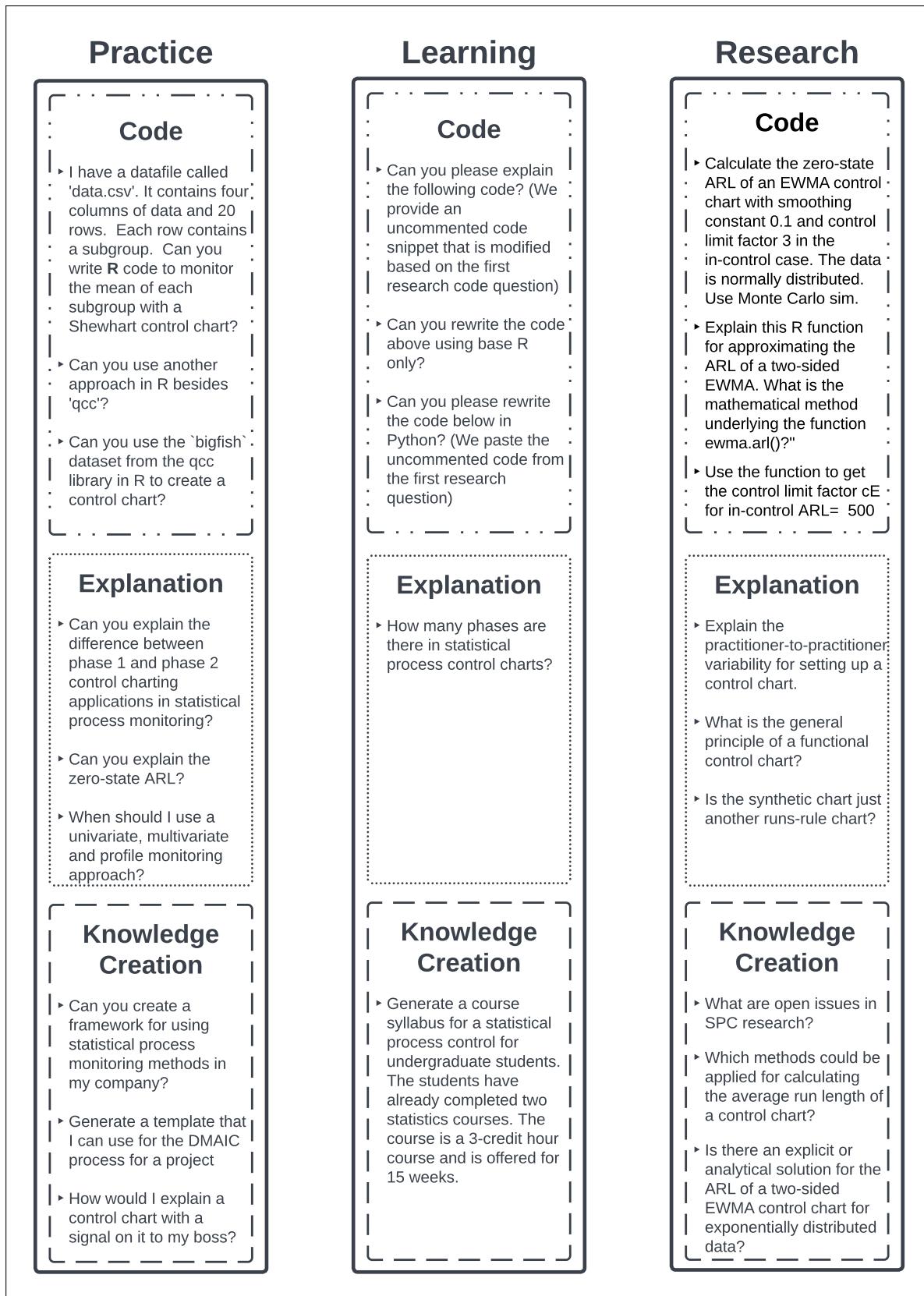


Figure 3: An overview of our study design.

an SPC course, and to determine open research issues in SPC. The exact prompts used appear in Figure 3 and Sections 5, 6, and 7.

Our team of experienced educators, authors, and statisticians evaluated the responses. Currently, ChatGPT can provide but not execute code. Therefore, our team copied and pasted the code into an appropriate environment (R or Python), attempted to run the code, and evaluated the correctness of the output. In cases where the code would not execute, our team corrected the code, explained the errors, and assessed the correctness of the output. We compared the ChatGPT responses to current textbooks and well-respected literature on the topics to evaluate the prompts related to explaining ideas and concepts. Similarly, we assessed the responses for knowledge creation and compared them to existing knowledge and similar assets for each domain area (practice, learning, and research).

Our methods are qualitative and are naturally colored by our own biases and experience in the area of SPC. A characteristic of generative AI models like ChatGPT is that responses to specific prompts are not repeated (which is why we have included many screenshots for documentation).

## 5 On the Use of ChatGPT by SPC Practitioners

We explore how ChatGPT can be used to augment coding, explain concepts, and create knowledge assets for practitioners. In the subsections below, we provide the results generated by our ChatGPT prompt, followed by our interpretation of “what worked” and “what failed”.

### 5.1 Code

First, we asked ChatGPT to write R code that can be used to create a Shewhart  $\bar{X}$  chart based on a ‘data.csv’ file containing 20 subgroups and four observations per subgroup. Figure 4 depicts our exact prompt and the corresponding code. From the output, ChatGPT incorrectly named the code chunk Python even though the code was output in R. ChatGPT selected a suitable R library, qcc, which can be used to construct a  $\bar{X}$  chart. The code assumes that the package is already installed on the user’s machine or that the user has a cursory knowledge of R to know that the `library()` function call would only work if the package has been installed in the past. Otherwise, the first line of code would return the following: “Error in `library(qcc)`: there is no package called ‘qcc’.” Then, the code assumes that the ‘data.csv’ is in the user’s working directory. In our estimation, both assumptions are reasonable given that the prompt specifically asked for R code. However, running the code results in the following error: “Error in `if (chart.all) { : argument is not interpretable as logical.`”

We attempted to understand why the code did not run. This process involved two steps. First, we tried eliminating the error and assigned the `chart` argument to a logical (e.g., `chart = TRUE`) based on the printed error message. This eliminated the error message but resulted in an incorrect chart (an individual chart instead of the desired  $\bar{X}$  chart, see Figure 5a). Second, we produced a correct chart, depicted in Figure 5b, by changing the `type` argument’s value from ‘`xbar.one`’ to ‘`xbar`’. We do not anticipate that the obtained error results from different package versions, as we checked earlier package releases from Scrucca (2004, 2017).

ChatGPT did not correctly assign arguments to the `qcc()` function: the values assigned to the `type` and `chart` arguments were incorrect. Moreover, the `subgroup.size` argument itself seems erroneous based on the source code for versions 1.0, 2.0, and 2.7 of the package. The argument’s correct name should have been `sizes`. It is also not needed based on the argument’s definition: “a value or a vector of values specifying the sample sizes associated with each group. For continuous data provided as a data frame or matrix, the sample sizes are obtained by counting the non-NA elements of each row. For ‘p’, ‘np’, and ‘u’ charts, the argument `sizes` is required.”

As a follow-up, we have examined whether ChatGPT can provide an alternate method for constructing the  $\bar{X}$  chart in R. The results are shown in Figure 6. Similar to the first prompt, the code starts by calling `library()` to load the required packages. The comments in the code highlight the steps needed to create a Shewhart chart using the `tidyverse` approach. However, similar to the previous code prompt, the resulting code is erroneous and will not produce correct results. Several issues were noted upon debugging the generated R code:

- (a) the attempted `group()` operation is incorrect since it assumes that every observation within a subgroup is in a separate row.
- (b) the object value in the subsequent line was never defined, which would result in an error due to “object ‘value’ not found”; and
- (c) neglecting the two previous errors, the use of the `geom_errorbar` is also incorrect in this instance.

 I have a datafile called 'data.csv'. It contains four columns of data and 20 rows. Each row contains a subgroup. Can you write R code to monitor the mean of each subgroup with a Shewhart control chart?

 Yes, here is an example of how to create a Shewhart control chart for the mean of each subgroup in R using the "qcc" package:

```
python
# Load the qcc package
library(qcc)

# Read in the data from the csv file
data <- read.csv("data.csv")

# Create a Shewhart control chart for the mean of each subgroup
qcc(data, type = "xbar.one", chart = "Shewhart", subgroup.size = 4)
```

In this example, `type = "xbar.one"` specifies that you want to create a control chart for the mean of each subgroup and `chart = "Shewhart"` specifies the type of control chart. The `subgroup.size` argument is set to 4 since you stated each subgroup has 4 rows.

Figure 4: A screenshot of ChatGPT’s response to reading the ‘data.csv’ file and writing an R code to monitor the mean of each subgroup with an Xbar Chart.

If we were to grade ChatGPT on consistency, it is unclear why two-sigma limits were used with `geom_errorbar` when the default 3-sigma limits were used in the ‘qcc’ function. Figure 7 presents our modified code and the resulting `ggplot`, with limits matching (with rounding error) those produced by the `qcc` package.

In the third and final practical code evaluation, we attempt to see if ChatGPT can detect erroneous requests. Given its apparent familiarity with the `qcc` package, we asked it to “can you use the ‘bigfish’ dataset from the `qcc` library in R to create a control chart?” Note that we made up this dataset name. Furthermore, we ensured that it does not exist in the package via the R command `data(package = ‘qcc’)`, which showed all the datasets within the `qcc` package (i.e., ‘bigfish’ was not in the printed list). In its initial output, ChatGPT utilized the exact solution approach presented in Practice Prompt #1. It modified that code (see Figure 8a) to load the non-existent dataset via the `data(bigfish)` function (which would have worked if the ‘bigfish’ dataset was indeed in the `qcc` package). In our estimation, ChatGPT’s use of `data(bigfish)` can be attributed to the use of reinforcement learning from human feedback (RLHF) to fine-tune the underlying GPT-3.5 model. While the RLHF approach was pioneered by OpenAI so that the LLMs are much better at following human intentions (Ouyang et al., 2022), our intentionally misleading question can be described as an adversarial attack on the model. However, the third line of code and the subsequent explanation were troubling since ChatGPT decided that the “`subgroup.size` argument is set to 5 since each subgroup in the ‘bigfish’ dataset contains 5 observations.” So we asked it a follow-up question that was not pre-planned. Our question and the ChatGPT explanation are depicted in Figure 8b. The “making up” of a seemingly accurate description based on a non-existent dataset within the package is also troubling since: (a) we could not find a “bigfish” dataset with such a description on the internet, and (b) the made-up description is believable (i.e., it clearly states why this dataset is included in the package, with a reasonable description of the number of subgroups and samples).

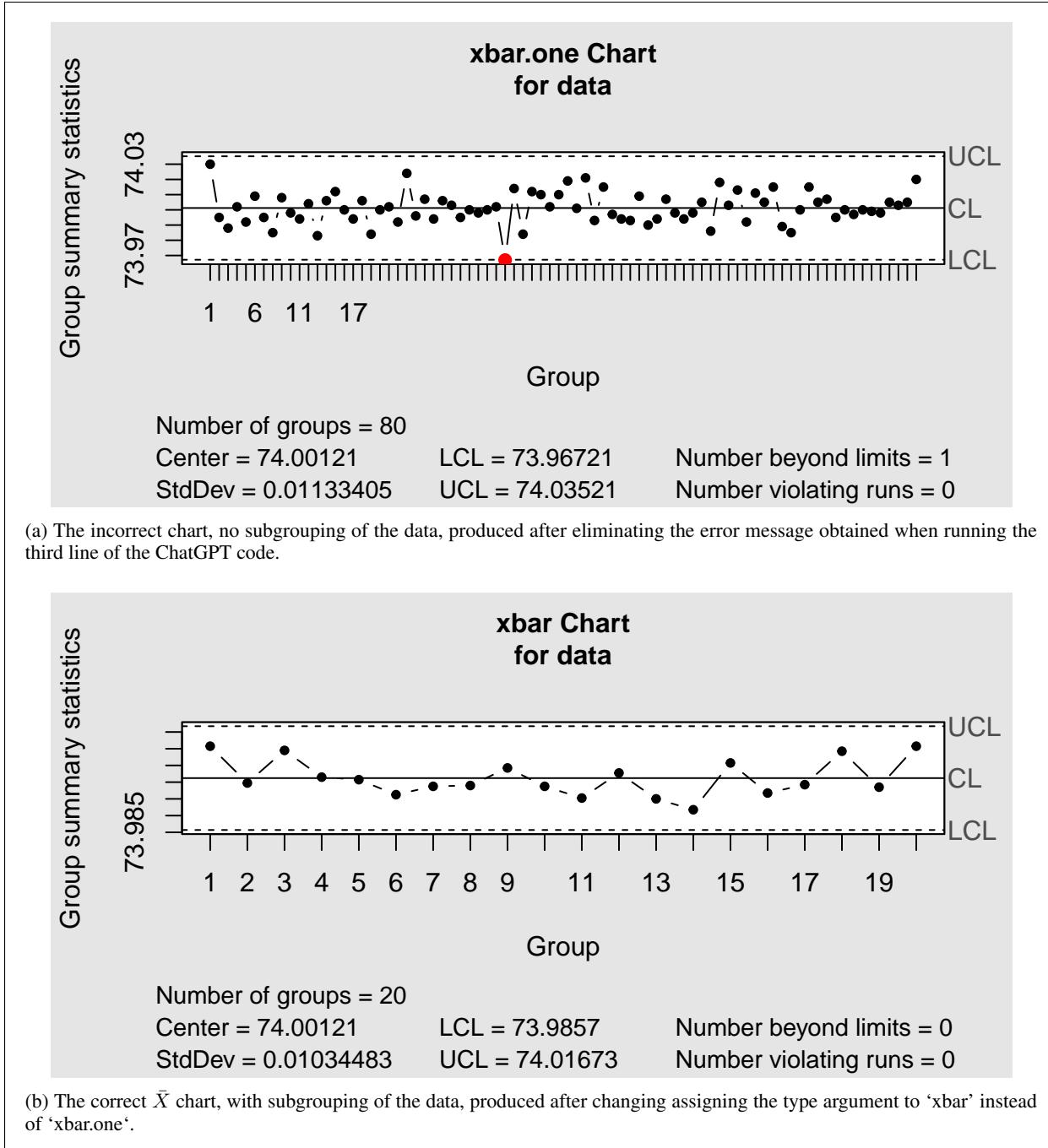


Figure 5: A comparison of the two produced charts, with slight modification to the arguments in the `qcc()` function used by ChatGPT.

## 5.2 Explanation

In this subsection, we examine ChatGPT’s ability to explain three concepts: (a) the difference between Phase 1 and Phase 2 control charting applications, (b) the zero-state ARL, and (c) when one should use a univariate, multivariate, and profile monitoring approach. The text-based responses for ChatGPT are depicted in Figure 9.

The first prompt asked ChatGPT to “explain the difference between phase 1 and phase 2 control charting applications in statistical process monitoring.” The response, shown in Figure 9, ChatGPT incorrectly stated that Phase 1 control charting is used to monitor the stability of the process over time. This is incorrect and more closely aligns with

 Can you use another approach in R besides 'qcc'.

 Yes, you can also create a Shewhart control chart in R using the "ggplot2" package. Here is  

```
SCSS
Copy code

# Load the ggplot2 and dplyr packages
library(ggplot2)
library(dplyr)

# Read in the data from the csv file
data <- read.csv("data.csv")

# Calculate the mean for each subgroup
data_means <- data %>%
  group_by(group = ceiling(row_number()/4)) %>%
  summarize(mean = mean(value))

# Create a Shewhart control chart for the mean of each subgroup
ggplot(data_means, aes(x = group, y = mean)) +
  geom_point() +
  geom_hline(yintercept = mean(data_means$mean), color = "red", linetype = "dashed")
  geom_errorbar(aes(ymin = mean - 2 * sd(data$value), ymax = mean + 2 * sd(data$value)))
  labs(x = "Subgroup", y = "Mean") +
  ggtitle("Shewhart Control Chart for the Mean of Each Subgroup")
```

In this example, the `group\_by` and `summarize` functions from the `dplyr` package are used to calculate the mean for each subgroup. The `ggplot` function creates the control chart, with points representing the mean of each subgroup and the red dashed line representing the overall mean. The error bars represent 2 standard deviations from the mean.

Figure 6: A screenshot of ChatGPT's response for our second practitioner-focused code prompt.

accepted definitions of Phase 2 of control charting. ChatGPT also responded that Phase 1 is focused on detecting changes in the process average; however, Phase 1 is not limited to evaluating the process average. Although it correctly stated that Phase 1 control charting is used to establish whether the process is in control and calculate control limits, it incorrectly stated that future observations are monitored over time in Phase 1. It is widely accepted that Phase 1 involves retrospective analysis of a process to evaluate process variability, checking for outliers, defining the in-control state of the process, developing a process model, estimating model parameters, and establishing a control chart for prospective monitoring (Jones-Farmer et al., 2014).

When defining Phase 2, ChatGPT begins incorrectly, stating that Phase 2 builds upon Phase 1 by monitoring the process variation. The definition of Phase 2 given by ChatGPT is similar to that of Phase 1 but focuses solely on

```

# Load the ggplot2 and dplyr packages
library(ggplot2)
library(dplyr)

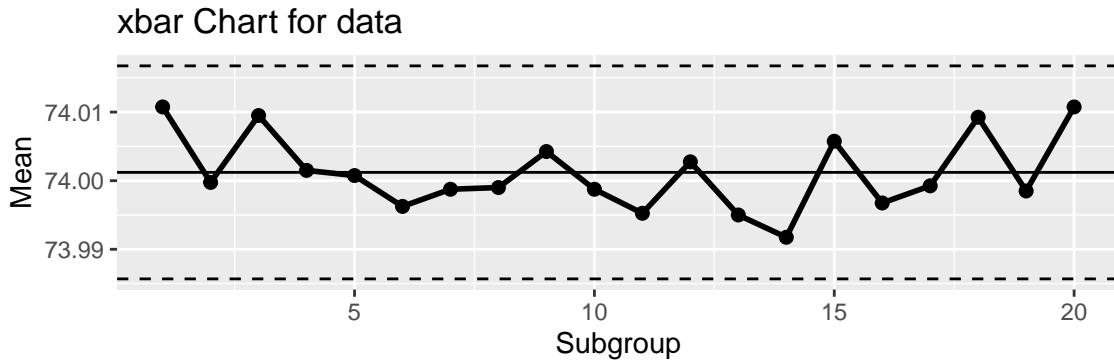
# Read in the data from the csv file
data <- read.csv("data.csv")

# Calculate the mean for each subgroup
data_means <- data |>
  group_by(group = row_number()) |>
  mutate(
    avg = mean(c(obs_1, obs_2, obs_3, obs_4)),
    R = max(c(obs_1, obs_2, obs_3, obs_4)) - min(c(obs_1, obs_2, obs_3, obs_4))
  ) |>
  ungroup()

# We do not recommend the use of ranges to estimate the standard deviation
# but we have used this approach here to reproduce the output from the qcc plot
# 0.729 is the control limit factor for the x-bar chart when subgroup size = 4
ggplot(data_means, aes(x = group, y = avg)) +
  geom_point(size = 2) +
  geom_line(size = 1) +
  geom_hline(yintercept = mean(data_means$avg), linetype = "solid") +
  geom_hline(yintercept = mean(data_means$avg) + 0.729*mean(data_means$R), linetype = "dashed") +
  geom_hline(yintercept = mean(data_means$avg) - 0.729*mean(data_means$R), linetype = "dashed") +
  labs(x = "Subgroup", y = "Mean") +
  ggtitle("xbar Chart for data")

```

(a) One possible approach, which capitalizes on the `ggplot2` and `dplyr` packages, to generate a *qcc*-equivalent  $\bar{X}$  chart while fixing the errors from the outputted ChatGPT code.



(b) The correct  $\bar{X}$  chart, with subgrouping of the data, produced after we significantly changed the ChatGPT code in Practice Prompt # 2.

Figure 7: Our `dplyr` and `ggplot2`-based code and the resulting  $\bar{X}$  chart, which is equivalent to that produced by the `qcc` package. Our 'data.csv' is a sample of the `pistonrings` dataset (we only keep the first 4 out of 5 observations for subsamples 1 – 20), and can be downloaded from the link provided in the *Online Materials* section.

monitoring variation. It is widely accepted that in Phase 2 of control charting, one uses the process model developed in Phase 1 to monitor prospectively for departures from the model. Phase 2 analysis is not limited to monitoring the process variability as implied by ChatGPT, but can be applied to any important process parameter identified in Phase 1. The final summary of the ChatGPT response states that Phase 1 control charting focuses on shifts in the average, while Phase 2 focuses on monitoring both the average and the variation. This is incorrect. For more details on Phase 1 and

 can you use the 'bigfish' dataset from the qcc library in R to create a control chart?

 Yes, you can use the `bigfish` dataset from the `qcc` package to create a control chart in R. 👍👎

Here is an example using the "qcc" package:

```
python
# Load the qcc package
library(qcc)

# Load the bigfish dataset
data(bigfish)

# Create a Shewhart control chart for the mean of each subgroup
qcc(bigfish, type = "xbar.one", chart = "Shewhart", subgroup.size = 5)
```

In this example, `type = "xbar.one"` specifies that you want to create a control chart for the mean of each subgroup and `chart = "Shewhart"` specifies the type of control chart. The `subgroup.size` argument is set to 5 since each subgroup in the `bigfish` dataset contains 5 observations.

(a) ChatGPT's 'explanation' of how the qcc package can be used to chart the 'bigfish' dataset.

 Can you describe the bigfish dataset?

 The `bigfish` dataset is included in the `qcc` package in R and contains data on the length of fish caught in the Bering Sea. The dataset consists of 34 subgroups, each with 5 observations of the length of a fish in centimeters. The `bigfish` dataset is used as a sample data set in many statistical process control (SPC) examples. It provides a simple and realistic scenario for illustrating how control charts can be used to monitor process stability over time and detect special causes of variation.

(b) ChatGPT makes up a somewhat compelling yet fake description of the 'bigfish' dataset.

Figure 8: ChatGPT is susceptible to simple adversarial attacks: A 'bigfish' story.

2 of process monitoring see, e.g., (Chakraborti et al., 2008; Jones-Farmer et al., 2014; Montgomery, 2020; Zwetsloot et al., 2023).

The second prompt asked ChatGPT to “explain the zero-state ARL.” The commonly accepted definition of the zero-state ARL is the average number of plotted statistics counted from the start of monitoring before a control chart signals (Crosier, 1986; Knoth, 2021). This is contrasted with the steady-state ARL, which measures the average number of plotted statistics until a signal is observed with counting beginning when the process is in steady state. A process is



Can you explain the difference between phase 1 and phase 2 control charting applications in statistical process monitoring?



Phase 1 control charting is used to monitor the stability of a process over time. It focuses on detecting shifts or changes in the process average and establishing whether the process is in control. In this phase, control limits are calculated based on historical process data, and future observations are plotted on the control chart to determine if they fall within the control limits. If they do, the process is considered to be in control.



Phase 2 control charting builds upon phase 1 by also monitoring the process variation. In this phase, control charts are used to monitor the process standard deviation and establish whether the process is consistent or predictable. Control limits for process variation are also calculated based on historical data, and future observations are plotted on the control chart to determine if they fall within these control limits. If they do, the process is considered to be consistent.

In summary, phase 1 control charting is focused on detecting shifts in the process average, while phase 2 control charting is focused on monitoring both the process average and process variation.

(a) ChatGPT's explanation of Phase 1 and Phase 2 control charting applications.



Can you explain the zero-state ARL?



The zero-state ARL (Average Run Length) is a measure of the performance of a statistical process control (SPC) chart, specifically control charts. It refers to the average number of observations that are taken before a false alarm or false signal occurs in a process that is actually in control. In other words, it measures the average number of observations required before the control chart signals a process shift, even though there is no actual shift in the process.



A lower zero-state ARL indicates that the control chart is more sensitive and is quicker to detect a process shift, while a higher zero-state ARL indicates that the control chart is less sensitive and requires more observations before detecting a process shift.

In the context of statistical process monitoring, zero-state ARL is important because it determines the frequency and speed at which false signals occur in a process. By controlling the zero-state ARL, process engineers can ensure that the control chart is sensitive enough to detect actual process shifts while avoiding false signals.

(b) ChatGPT's explanation of the zero-state ARL.

defined to be in steady-state if “the process has been in control long enough for the effect of the starting value [of the chart statistic] to become negligible” (Lucas and Saccucci, 1990).

 When should I use a univariate, multivariate and profile monitoring approach?

 Univariate, multivariate, and profile monitoring are three different statistical process control (SPC) methods used for monitoring the quality of a process. thumb up thumb down

Univariate monitoring is used when the process produces a single output measurement, such as weight or length. Univariate monitoring uses control charts to monitor the process mean and variability, and detect any significant changes in the process over time.

Multivariate monitoring is used when the process produces multiple outputs or variables, such as the pH, temperature, and pressure of a chemical reaction. Multivariate monitoring uses techniques such as principal component analysis (PCA) and partial least squares (PLS) to model the relationship between the variables and detect any changes in the process.

Profile monitoring is used when the process produces a product with a complex shape, such as a molded plastic part. Profile monitoring uses techniques such as multivariate statistical process control (MSPC) to monitor the shape and dimensions of the product and detect any changes in the process.

In summary, univariate monitoring is used for processes with a single output, multivariate monitoring is used for processes with multiple outputs, and profile monitoring is used for processes that produce a complex shaped product. The appropriate method to use depends on the type of data generated by the process and the goals of the process monitoring.

(c) ChatGPT's explanation of when to use univariate, multivariate, and profile monitoring approaches.

Figure 9: ChatGPT's response to our three explanatory practitioner prompts.

When given the prompt to explain the zero-state ARL, ChatGPT provided a partially correct answer but either ignored or misinterpreted the qualifier “zero-state”. The basic definition given by ChatGPT is a mostly correct definition of the in-control (not zero-state) ARL as the “average number of observations that are taken before a false alarm or a false signal occurs in a process that is actually in control.” The accepted definition of the ARL is the average number of *plotted statistics* observed before a signal is observed. The use of the term *plotted statistic* in place of *observations that are taken* is more general and can be applied to subgrouped data. Adding to this definition the average number of plotted statistics observed before a *false* signal is observed provides a correct definition of the in-control ARL. ChatGPT then restates the definition: “it measures the average number of observations required before the control chart signals a process shift, even though there is no actual shift in the process.” Once again, this is a near-correct description of the in-control ARL, but there is no mention of when counting begins to indicate zero-state. Thus, we can likely conclude that ChatGPT misinterpreted the *zero-state* qualifier for the ARL as *in-control*.

In the next statement describing the ARL, ChatGPT correctly stated that a lower ARL indicates a chart that is quicker to detect a process change, and a chart with a higher ARL requires more observations to detect a change. In addition, ChatGPT correctly contextualized the ARL as a tool for process engineers to understand the sensitivity of a chart to changes, the frequency of signals, and the usefulness of balancing the detection of changes with avoiding false signals.

The third prompt provided to ChatGPT asked “When should I use univariate, multivariate, and profile monitoring?” ChatGPT correctly noted that univariate monitoring is used to monitor a single process measure over time. The answer given, however, is specific to variables data, noting that one monitors the mean and variance with univariate control charts. A more general answer would include reference to, e.g., proportions, counts, and location measure other than the mean.

The advice for multivariate monitoring is also correct, noting that it is used for monitoring multiple process variables. Using a simple example of monitoring the pH, temperature, and pressure of a chemical reaction, ChatGPT suggests that one would use methods such as principal components analysis (PCA) or partial least squares (PLS) for multivariate monitoring. While these methods are useful for monitoring high-dimensional processes, the most widely used multivariate control chart, Hotelling's  $T^2$  chart was not mentioned.

ChatGPT also correctly identified the context for profile monitoring to monitor a shape such as that of a molded plastic part. It qualified this, however, to be a *complex shape*. In actuality, profile monitoring can be used for both simple and complex shapes, and monitoring linear profiles is common in practice (Kim et al., 2003). ChatGPT states that multivariate statistical process control is used to monitor the shape and dimensions of the profile. This is a correct, but very general statement. For reviews of the research on profile monitoring, see, (Maleki et al., 2018; Noorossana et al., 2013; Woodall, 2007)

### 5.3 Knowledge Creation

In this section, we ask ChatGPT to provide descriptions that can be used in reporting SPC concepts in practice. We asked it the following questions:

- (a) Can you create a framework for using statistical process monitoring methods in my company?
- (b) Generate a template that I can use for the DMAIC process for a project.
- (c) How would I explain a control chart with a signal on it to my boss?

The generated responses were reasonable, similar to our discussion in Section 5.2.

For question (a), ChatGPT presented a framework made of the following seven steps: (i) define the process, (ii) collect data, (iii) choose a control chart, (iv) establish the control limits, (v) monitor the process, (vi) detect and respond to out-of-control conditions, and (vii) continuously evaluate and improve. For each step, it provided a 1-2 sentence explanation and concluded that the framework might need to be modified based on the company's requirements. Similarly, for question (b), ChatGPT identified 2-3 tasks that should be completed within each step of the DMAIC process. Its conclusion followed the previous question, stating that this is a general framework that may need to be tweaked based on the application domain. Note that we do not provide snapshots of its responses here for conciseness. The interested reader should refer to our *Supplementary Materials* for the recordings of the ChatGPT answer.

ChatGPT provided a compelling answer to the last question (see Figure 10). While we did not explicitly ask for steps, the AI likely learned from the two previous questions and provided an excellent breakdown of the question. Its introductory paragraph provided working definitions for control charts and signals, and then it highlighted five steps that should be followed when explaining a signal to one's boss. It wrapped up by emphasizing the need to communicate the principles behind a control chart and signal effectively while ensuring the significance and the steps taken to resolve the signal.

While we appreciate the points made by ChatGPT in its answer, we note that the output lacks sufficient statistical detail. This can partially be attributed to the generalized nature of our input prompt/question. However, it also highlights the differences between a chat with an AI versus a statistical consultant. In the AI case, the questions are only asked by the human interacting with ChatGPT. However, in a human-to-human interaction, the statistical consultant would have likely asked several questions to tailor their response to the audience. These questions would have likely included the following:

- What are the boss's title and training background?
- Was this signal observed in Phase I or Phase II?
- What type of control chart was used?
- How were the control limits determined?
- What is the application domain (e.g., in public health surveillance, the control chart does not typically reset after a Phase II signal)?
- Can the process be stopped until a thorough investigation of the process occurs?

From these answers, the consultant would be able to generate a more precise response. When interacting with an AI chatbot, the onus is on the human to provide this context upfront or ask the AI to tweak its answer by supplementing its responses with additional information.

 How would I explain a control chart with a signal on it to my boss?

 A control chart is a statistical tool that is used to monitor the stability of a process over time. It consists of a line graph that represents the process performance, with control limits set above and below the line to indicate when the process is in control or out of control. A signal on the control chart is a visual representation of an unusual data point or a deviation from the expected process performance.

To explain a control chart with a signal to your boss, you can use the following steps:

1. Provide context: Explain the purpose of using a control chart and the process that it is monitoring.
2. Explain the control chart: Show the control chart to your boss and explain how it works, including the line graph that represents the process performance, and the control limits that indicate when the process is in control.
3. Highlight the signal: Show the signal on the control chart and explain that it represents an unusual data point or a deviation from the expected process performance.
4. Discuss the significance: Explain the significance of the signal, including any potential impact on the process, the customer, or the business.
5. Outline next steps: Explain the next steps that will be taken to investigate and address the signal, including any necessary data analysis, root cause analysis, or corrective actions.

By using this structure, you can clearly and effectively communicate the key information about the control chart and the signal to your boss, ensuring that they understand the significance of the issue and the steps that will be taken to resolve it.

Figure 10: ChatGPT’s answer on how to explain a control chart signal to your boss.

## 6 Learning SPC Fundamentals with ChatGPT

### 6.1 Code

To examine whether ChatGPT can accelerate learning in the context of coding, we have designed three coding tasks. These three coding tasks relate to understanding code, providing an alternative approach to solving a problem with code, and translating code from one language to another. These questions and ChatGPT’s corresponding response are detailed in the following paragraphs.

In our first question, we provided ChatGPT with an uncommented version of our code prompt from Figure 7a and asked to explain the code. This question was asked in a new chat window so that no context from our previous discussions is given. ChatGPT provided a comprehensive explanation of our code, as shown in Figure 11. It clearly captured the different steps and provided an explanation of the different functions. It even recognized that this code is used for the creation of an  $\bar{X}$  control chart (possibly based on the inputs from the `ggtitle()` function). One missing aspect from its explanation was to highlight what the three horizontal lines stand for, i.e., that the `geom_hline()` was used to draw the center line (with the solid line) and the lower control limit (LCL)/upper control limit (UCL) with the dashed lines. Overall, we are pleasantly surprised by its explanation.

In our second question, we asked ChatGPT to “rewrite the code above using base R only”. Figure 12a presents the output from ChatGPT with our minor modification in the second line. If we were to ignore our change momentarily,

This is an R script that creates an Xbar control chart.

- The script first loads the `ggplot2` and `dplyr` libraries.
- The data is read from a `.csv` file into a data frame named "data".
- The data is then processed with the "dplyr" package to create a new data frame named "data\_means".
  - The data is grouped by the row number, with each row representing a subgroup.
  - The mean of the observations in each subgroup (`obs_1`, `obs_2`, `obs_3`, and `obs_4`) is calculated and stored in a new column named "avg".
  - The range of the observations in each subgroup is calculated and stored in a new column named "R".
  - The data is then ungrouped.
- The processed data is then plotted using the `ggplot2` library:
  - A scatter plot with points and a line connecting the points is created, using "group" as the x axis and "avg" as the y axis.
  - Three horizontal lines are added to the plot: the first represents the mean of the "avg" column, and the other two represent the mean plus and minus 0.729 times the mean of the "R" column.
  - Labels are added to the x and y axes, and a title is added to the plot.

Figure 11: ChatGPT's explanation of an uncommented version of the code shown in Figure 7a. We provided the code in a new chat window so that ChatGPT would have no context for our previous conversations.

one could observe that ChatGPT was able to successfully: (a) replace the tidyverse-heavy `group_by`, `mutate`, and `ungroup` style into an appropriate base R coding style with the use of `rowMeans`, and `apply` (with a custom function to define the range instead of using the built-in `range` function); and (b) produce a base R equivalent plot (see Figure 12b) to our earlier `ggplot2`-approach.

ChatGPT's use of the `group = row_number(data)` was incorrect for two main reasons. First, the `row_number()` function comes from the `dplyr` package, which is not from base R and was not loaded in its generated prompt. Second, the use of `row_number` in our earlier code was appropriate/correct due to the `group_by()` operator. However, without grouping, the implementation is equivalent to the base R: `rank(ties.method = "first")`. The input to the `rank` function should be a vector; hence, the output from the ChatGPT code is a ranking of 80 values ( $20 \times 4$ , i.e., our 20 subgroups, each containing four observations). With this incorrect numbering/ranking and R's defaults of plotting data from left to right, the lines connecting the subgroups no longer capture the correct time order, and the resulting plot would be incorrect. Obviously, this error was mitigated by using `row.names()` in our modified code snippet, which produced the correct plot. Despite this minor error, we were impressed with ChatGPT's output to this prompt.

In our third question, we asked ChatGPT to rewrite the code from Figure 7a in Python. To ensure that it is not affected by the slightly incorrect base R code, our prompt was "Can you please rewrite the code below in Python?" with the pasted code being the uncommented code version of Figure 7a. The output from this prompt is shown in Figure 13a, and the plot generated by running this code in a Python environment is shown in Figure 13b. In this case, the generated code worked directly with no modifications. ChatGPT was able to identify the appropriate libraries/functions to rewrite our R code into Python. ChatGPT's success was likely possible since our original code was based on well-known packages that are well-documented online. Nevertheless, we think that ChatGPT's success here (even if it is limited to well-known packages) would reduce the burden of learning new programming languages and allow sharing (some) code quickly in multiple programming languages.

```

data <- read.csv("data.csv")

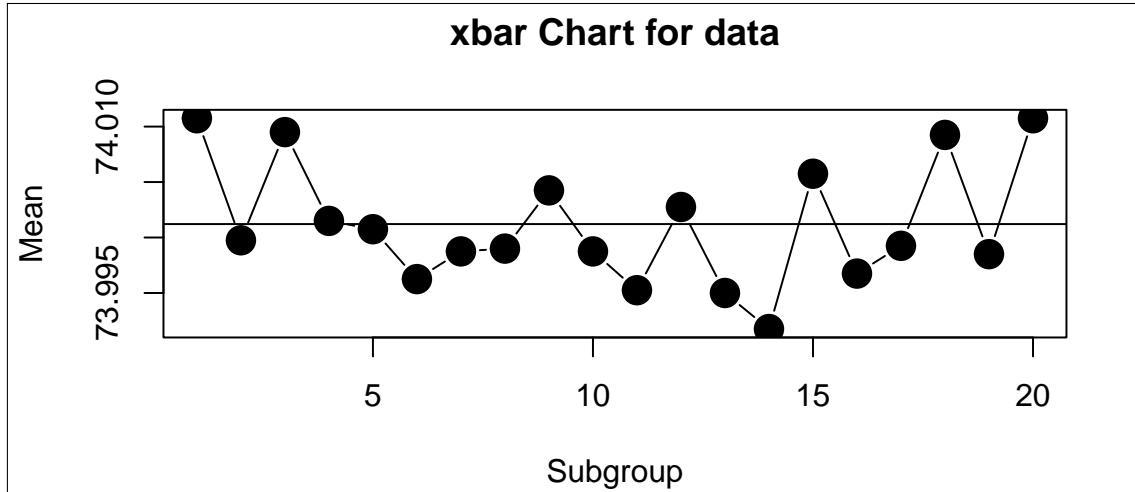
group <- row.names(data) # Changed from ChatGPT's incorrect: group <- row_number(data)

data_means <- data.frame(
  group = group,
  avg = rowMeans(data[, c("obs_1", "obs_2", "obs_3", "obs_4")]),
  R = apply(data[, c("obs_1", "obs_2", "obs_3", "obs_4")], 1,
            function(x) max(x) - min(x))
)

plot(data_means$group, data_means$avg, type = "b", pch = 19, cex = 2,
      xlab = "Subgroup", ylab = "Mean")
abline(h = mean(data_means$avg), lty = 1)
abline(h = mean(data_means$avg) + 0.729 * mean(data_means$R), lty = 2)
abline(h = mean(data_means$avg) - 0.729 * mean(data_means$R), lty = 2)
title(main = "xbar Chart for data")

```

(a) The base R code generated by ChatGPT in lieu of Figure 7a. The darker second row highlights a minor change to ChatGPT's code where we used the base R function `row.names` instead of the incorrect `dplyr` function `row_number`.



(b) The correct  $\bar{X}$  chart produced with the minor modification of the ChatGPT code. Note that the UCL and LCL overlap with the existing boundaries for the chart. Hence, to show them, the reader can set the `ylim` argument in the plot to an appropriate value (e.g., `ylim = c(73.98, 74.02)`.)

Figure 12: The ChatGPT base R code (with one minor change) and its associated  $X$  chart.

## 6.2 Explanation

To examine ChatGPT's to provide learning-related explanations, we asked, "How many phases are there in statistical process control charts?" We intentionally worded the question similar to the first Practice-based explanation question (shown in Figure 9a), hoping for a two-phase answer. However, ChatGPT responded by stating that SPC charts have five phases (Figure 14). While we did not expect such an answer, and it is unlikely to be given by an SPC expert, it is not incorrect. One can argue that these are steps followed when implementing an SPC monitoring plan.

To ensure that we were not reading too much into ChatGPT's answer, we prompted it to "explain the five phases above in greater detail." Figure 15 depicts ChatGPT's explanation. To improve ChatGPT's explanation, we would have augmented its phases with the following (each bullet corresponds to one phase):

1. Included a discussion of measurement systems analysis to ensure that the gauges used in process measurement are repeatable and reproducible. The reader is referred to Montgomery (2020) for a detailed introduction to measurement system analysis.

```

import pandas as pd
import matplotlib.pyplot as plt

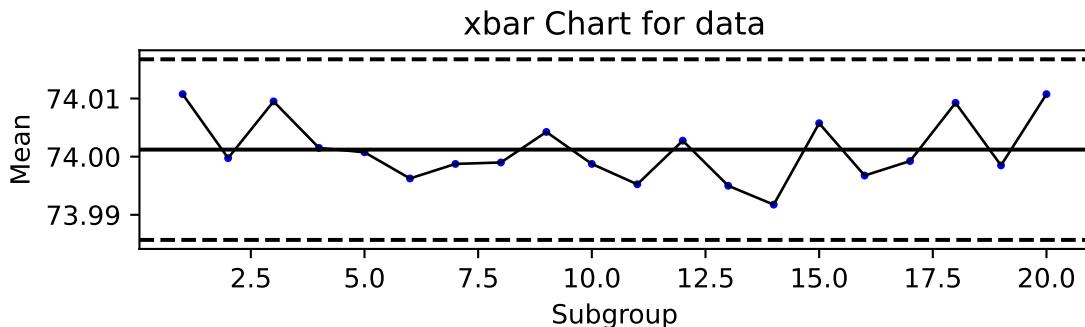
data = pd.read_csv('data.csv')
data['group'] = data.index + 1
data_means = data.assign(
    avg=data[['obs_1', 'obs_2', 'obs_3', 'obs_4']].mean(axis=1),
    R=data[['obs_1', 'obs_2', 'obs_3', 'obs_4']].max(axis=1) - data[['obs_1', 'obs_2',
        'obs_3', 'obs_4']].min(axis=1)
)

plt.plot(data_means['group'], data_means['avg'], 'bo', markersize=2)
plt.plot(data_means['group'], data_means['avg'], 'k-', linewidth=1)

plt.axhline(data_means['avg'].mean(), color='k', linestyle='solid')
plt.axhline(data_means['avg'].mean() + 0.729 * data_means['R'].mean(), color='k',
    linestyle='dashed')
plt.axhline(data_means['avg'].mean() - 0.729 * data_means['R'].mean(), color='k',
    linestyle='dashed')
plt.xlabel('Subgroup')
plt.ylabel('Mean')
plt.title('xbar Chart for data')
plt.show()

```

(a) The Python code generated by ChatGPT in lieu of Figure 7a.

(b) The correct  $\bar{X}$  chart produced with ChatGPT using the pandas and matplotlib libraries.Figure 13: The ChatGPT generated Python code and its associated  $\bar{X}$  chart.

 How many phases are there in statistical process control charts?

 There are typically 5 phases in Statistical Process Control (SPC) charts: thumb up thumb down

1. Understanding the process
2. Defining the process and specifying control limits
3. Collecting data and plotting the control chart
4. Analyzing the data and interpreting the chart
5. Taking action based on the chart analysis.

Figure 14: ChatGPT's statement that there are five phases in SPC control charts.

2. ChatGPT's explanation of this step seems to be more centered around Phase 1 control chart applications, where the goals are to ensure that the process is in-control, estimate the process parameters and establish control limits (see our discussion in Section 5.2). To make the answer actionable, we would have expected a discussion of what to do if there is a plotted statistic outside the control chart in this phase.
3. This phase corresponds to Phase 2, where the process is monitored in real-time (or in fixed intervals if 100% sampling is not employed).
- 4-5. Here, we do not have anything to add to the ChatGPT output. In our opinion, the answers for phases 4 and 5 are sufficient.

The screenshot shows a conversation between a user and ChatGPT. The user asks, "Explain the five phases above in greater detail". ChatGPT responds with a detailed explanation of the five phases of Statistical Process Control (SPC) control charts, numbered 1 through 5. The response is presented in a large text block. Below the text, a note states: "These five phases are an iterative process that is repeated over time to continuously monitor and improve the process performance." There are like and dislike icons at the top right of the response area.

Explain the five phases above in greater detail

Sure, here's a detailed explanation of the five phases of Statistical Process Control (SPC) control charts:

1. Understanding the process: In this phase, the focus is on understanding the process being studied, including its inputs, outputs, and how they are related. This requires a deep understanding of the process itself, its variables, and their relationships.
2. Defining the process and specifying control limits: In this phase, the process is defined more formally and the control limits are specified. The control limits are calculated based on statistical analysis of the data collected from the process and represent the upper and lower bounds for process performance.
3. Collecting data and plotting the control chart: In this phase, data is collected from the process and plotted on a control chart. The control chart is a graphical representation of the process performance over time, and helps to identify trends, patterns, and special cause variations.
4. Analyzing the data and interpreting the chart: In this phase, the data on the control chart is analyzed to determine if the process is in control or out of control. A process is considered to be in control if the plotted data falls within the control limits, and out of control if it falls outside the limits.
5. Taking action based on the chart analysis: In this phase, actions are taken based on the analysis of the control chart. If the process is in control, no action is typically required. However, if the process is out of control, corrective actions must be taken to bring the process back into control. These actions may include changes to the process inputs, modifying the process, or other interventions.

These five phases are an iterative process that is repeated over time to continuously monitor and improve the process performance.

Figure 15: ChatGPT's “in-depth” explanation of the five phases in SPC control charts.

### 6.3 Knowledge Generation

We asked ChatGPT to “Generate a course syllabus for a statistical process control for undergraduate students. The students have already completed two statistics courses. The course is a three-credit hour course and is offered for 15

weeks.” In our prompt, we ensure that ChatGPT knows the target audience, i.e., undergraduate students who have already completed two statistics courses and the course duration of 15 weeks (typical in the U.S. semester system). Its output, which was too long to include in one figure, consisted of several sections:

- (a) introductory syllabus material: which included the course title, prerequisites, duration, number of credit hours, and to be filled-in information about the instructor, their email, and office hours.
- (b) course description: “This course provides an in-depth study of statistical process control (SPC) techniques and their applications in industry. Students will learn how to use statistical methods to monitor, control, and improve process quality in manufacturing and service industries. The course will cover both theoretical and practical aspects of SPC and include hands-on experience using statistical software.”
- (c) learning outcomes:
  - “Describe the purpose and principles of statistical process control.”
  - “Choose and apply appropriate statistical methods for process control.”
  - “Use statistical software to perform process monitoring and control.”
  - “Interpret process control charts and identify special cause variation.”
  - “Evaluate process stability and determine appropriate process improvement actions.”
- (d) To-be-filled-in textbook, grading, and class policies.
- (e) Schedule of topics, which consisted of:
  - “Week 1: Introduction to SPC”
  - “Week 2-3: Control charts for variables”
  - “Week 4-5: Control charts for attributes”
  - “Week 6-7: Process capability analysis”
  - “Week 8-9: Process improvement using SPC”
  - “Week 10-11: Design of experiments”
  - “Week 12-13: Multi-vari study” (unclear if this meant multivariate SPC or the multi-vari chart)
  - “Week 14-15: Group project and presentation”

We should note that the generated course description, learning outcomes, and schedule of topics were reasonable (can be adopted without major changes). We were impressed with this ChatGPT answer, which undoubtedly benefited from the many SPC syllabi found online and were likely included in its training. The interested reader is referred to our online supplementary materials for a link to the recording of ChatGPT’s generated answer.

## 7 On the Use of ChatGPT in SPC Research

### 7.1 Code

To examine ChatGPT’s ability to generate code that can supplement potential SPC research projects, we provided ChatGPT with the following prompt “calculate the zero-state ARL of an EWMA control chart with smoothing constant 0.1 and control limit factor 3 in the in-control case. The data is normally distributed. Use Monte Carlo simulation.” ChatGPT responded with a Python-generated code, which we reproduce in Figure 16.

Similar to the previous Python-generated code chunk in Figure 13a, the above Python code runs without errors and produces a result. However, unlike the previous code chunk, this produced answer is incorrect. In particular, our testing of the `simulate_ewma()` function resulted in values of 260.8, 255.8, 262.9, and 260.2. The correct result should be close to 842.1, 1035.1 (if we account for the skipped standardization), or 371 (if we corrected for interchanging `alpha` and `1-alpha`). In our diagnosis of the function, we identified two major flaws. First, the run-lengths are truncated at  $n = 1000$ . In about 40% of the Monte Carlo simulations, the truncation results in a zero run length making the final result small. Second, the control limit incorporates the sample standard deviation of the underlying normal sample, which means that future/unseen data is used for establishing the control limit. We also observed minor flaws, including (a) the EWMA series is initialized with the first observation, which is unusual in the SPC literature (perhaps, ChatGPT confounded the initialization with the simple exponential smoother in time-series analysis), (b) interchanging `alpha` and `1-alpha`, and (c) missing the standardization of the control limit, which is typically written in multiples of the EWMA standard deviation.

In our second prompt, we asked to “explain the following R function for approximating the ARL of a two-sided EWMA control chart for monitoring the mean of normally distributed random variables ....” ChatGPT produced a verbose answer, which we do not reproduce here for space limitations (see our supplementary *Online Materials* for a video recording of the output). However, we followed up with “what is the name of the mathematical method underlying the function `ewma.arl1()`?” The corresponding answer was remarkably short and notably close to the

```

import numpy as np
import matplotlib.pyplot as plt

def ewma(x, y, alpha):
    return (1 - alpha) * x + alpha * y

def simulate_ewma(n, alpha, k, num_simulations):
    arls = []
    for i in range(num_simulations):
        x = np.random.normal(0, 1, n)
        ewma_stat = np.zeros(n)
        ewma_stat[0] = x[0]
        for j in range(1, n):
            ewma_stat[j] = ewma(x[j], ewma_stat[j-1], alpha)
        control_limit = k * np.std(x)
        run_length = 0
        for j in range(n):
            if np.abs(ewma_stat[j]) > control_limit:
                run_length = j + 1
                break
        arls.append(run_length)
    return np.mean(arls)

n = 1000
alpha = 0.1
k = 3
num_simulations = 10000

arl = simulate_ewma(n, alpha, k, num_simulations)
print("Zero-state ARL:", arl)

```

Figure 16: The ChatGPT-generated Python code in response to our request to calculate the zero-state ARL for the EWMA chart for normally distributed data via Monte Carlo simulation.

complete answer “Markov chain approximation”. The code and the corresponding ChatGPT response are presented in Figure 17.

As a final coding question, we asked ChatGPT to “use the aforementioned function to get the control limit factor  $c_E$  for an in-control ARL of 500.” Note that this is the inverse task to that provided by our `ewma.arl()` function since we want to determine the control limit for a given ARL. ChatGPT’s response is presented in Figure 18. While the solution steps are generally correct, the code delivers “`f(lower) is NA`” in its current form. The adverse interval choice in the `uniroot()` function prevents solution generation. Both the lower and upper limits for the `interval` argument need to be changed. The lower limit has to be positive, otherwise the underlying function `ewma.arl()` fails to deliver a suitable result (we could obviously hard-code the ARL to equal 1 if the lower interval argument is zero). In our opinion, lifting the lower interval limit within the `uniroot()` function to a positive value is more appropriate. Furthermore, the upper limit has to be substantially smaller than 100. For  $c_E = 5$ , the function `ewma.arl()` would return an  $ARL > 2,000,000$ . Hence, an upper limit substantially larger than 5 is both unnecessary and creates numerical difficulties for  $c_E$  values larger than 10 (with ARL values larger than  $10^{16}$ ). Setting `interval = c(0.1,5)` results in 2.8156, which produces the correct outcome.

## 7.2 Explanation

Here, we consider one of our explanation tasks and refer to the supplemental material for the other two. We asked ChatGPT to “Explain the practitioner-to-practitioner variability for setting up a control chart”. We picked this task because it became a popular and handy notion for describing the uncertainty introduced by estimating population parameters for setting up the control limits of control charts (Jardim et al., 2020; Saleh et al., 2015; Zhang et al., 2013). ChatGPT’s response is presented in Figure 19.

Similar to previous explanation tasks (in Sections 5 and 6), the generated response is somewhat long and imprecise. Specifically, ChatGPT presented five factors, which share a common feature; all deal with differences on the *method*

```

ewma.arl <- function(lam, cE, mu, z0=0, r=50) {
  cE <- cE*sqrt(lam/(2-lam))
  ii <- (-r:r)
  w <- 2*cE/(2*r+1)
  qij <- function(i,j) pnorm((j*w-(1-lam)*i*w+w/2)/lam-mu) -
    pnorm((j*w-(1-lam)*i*w-w/2)/lam-mu)
  Q <- outer(ii, ii, qij)
  one <- array(1,2*r+1)
  I <- diag(1,2*r+1)
  ARL <- solve(I-Q, one)
  arl <- ARL[r+round((z0+w/2)/w)]
  arl
}

```

(a) The inputted `ewma.arl()` R function, which is to be explained by ChatGPT.



What is the name of the mathematical method underlying the function `ewma.arl()`?



The mathematical method underlying the function `ewma.arl()` is Markov Chain analysis. thumb up thumb down

(b) ChatGPT's response about the mathematical method underlying the function `ewma.arl()`.

Figure 17: Our input `ewma.arl()` function and ChatGPT's naming of the underling model.

level, i.e., chart type, subgroup design, techniques to calculate the limits, dealing with outliers, and choice of software. While we agree that these factors are important and will drive different results, ChatGPT's answer ignores the context for which the “practitioner-to-practitioner” variability is used in the SPC literature. In fact, the “practitioner-to-practitioner” variability refers to the variation that occurs with a fixed configuration of the five aforementioned factors, i.e., the variation results from multiple implementations of the same procedure on the same data-generating process. The interested reader is referred to Zhang et al. (2013) for a detailed introduction to “practitioner-to-practitioner” variability, and how the standard deviation of the average run length (SDARL) metric can be used to capture such variability.

### 7.3 Knowledge Generation

Inspired by the TEDxBoston talk titled “what we learned from 5 million books” (Michel and Aiden, 2011), we were curious to see what is ChatGPT’s answer to open issues in our field given that it likely “read” and “can recall” more SPC research papers than most (if not all) of us. Hence, we asked, “What are open issues in statistical process control research?” Its generated response, depicted in Figure 20, highlighted seven themes that included (a) incorporating big data and machine learning techniques, (b) online/real-time monitoring solutions where 100% sampling is employed, (c) reducing the need for non-normality, and (d) applications to new domains.

While the answer is reasonable, the real value, in our opinion, is the potential to use generative AI tools for research idea generation/validation. While this is likely to be only valid on a high level, this can be viewed through a similar lens to bibliometric approaches that identify literature clusters/themes based on keywords and citations (e.g., see Megahed, 2019). Furthermore, the ideas can potentially be “stale” as the ChatGPT/GPT-3.5 model “finished training in early 2022” (OpenAI, 2022a). It is important to note that the obsolescence of responses may not be an issue in future LLM generations since there is a growing body of research examining the combination of external knowledge sources/APIs with LLMs (Karpas et al., 2022; Schick et al., 2023).

 Here's an example of how to use the `ewma.arl` function to get the control limit factor `cE` for an in-control ARL of 500 with a smoothing constant of 0.1, a mean of 0, and a starting value of 0 for the EWMA statistic:

```
r
ewma.arl.target <- 500
ewma.arl.optimize <- function(cE) {
  ewma.arl(lam = 0.1, cE = cE, mu = 0, z0 = 0) - ewma.arl.target
}

cE <- uniroot(ewma.arl.optimize, interval = c(0, 100))$root
cE
```

The function `uniroot` is used to find the root of the equation `ewma.arl.optimize(cE) = 0`, which corresponds to an in-control ARL of 500. The `interval` argument specifies the search interval, which in this case is `c(0, 100)`. The root of the equation can then be found using `uniroot(ewma.arl.optimize, interval = c(0, 100))\$root`.

Note that this example is for demonstration purposes only, and the accuracy of the solution depends on the choice of the interval, the accuracy of the approximation provided by the `ewma.arl` function, and the distributional assumptions of the data.

Figure 18: ChatGPT’s code to get the control limit factor CE for an in-control ARL of 500.

## 8 Discussion and Conclusions

### 8.1 Our Perspective on ChatGPT’s (GPT-3.5) Ability to Augment SPC Practice, Learning, and Research

In this expository paper, we attempted to examine whether generative AI models, such as ChatGPT, can inform SPC practice, learning, and research. To inform our understanding, our research team has engineered several prompts to capture some tasks that SPC practitioners, learners, and researchers may use to interact with ChatGPT. Our prompts within each grouping were divided into code, explanation, and knowledge generation subgroups to examine ChatGPT’s utility/ability in performing these different tasks. From our results in Sections 5-7, we have several general observations on ChatGPT’s ability to help the different SPC stakeholders. It is important to note that our observations are based on the *ChatGPT Jan 30 Version (Free Research Preview)*, which we used in all of our interactions with ChatGPT.

First, one feature of generative AI models such as ChatGPT is that the answers to a given prompt are not repeated. We highlighted this feature in the introductory paragraph of Section 2. We were curious to see how this feature translates when interacting with ChatGPT. In our initial experimentation, our research team members used a sample of our prompts in the same order under similar conditions (i.e., we started a new conversation for each subtopic). The obtained outputs differed not only in terms of notation/wording but also in content. For example, our “open issues in SPC research” prompt generated responses that included *design of experiments* and others that ignored this. Furthermore, the number of open issues differed by the response. In our opinion, this can be considered both a benefit and a limitation. On the one hand, the changing content can be viewed as a feature for prompts where there should not be one single answer (e.g., our open issues question or the notations used in the code). On the other hand, it is a possible deterrent in situations when a “ground truth” is needed. A detailed examination of singular-answer-prompts should be explored in future research.

 Explain the practitioner-to-practitioner variability for setting up a control chart.

 Control charts are used in quality control to monitor the process performance over time and detect any unusual patterns or deviations from the expected behavior. The process of setting up a control chart can vary based on the expertise and preferences of the practitioner.

Practitioner-to-practitioner variability in setting up a control chart can stem from several factors, including:

1. Choice of chart type: Different practitioners may choose different types of control charts based on the data and the process being monitored. For example, some practitioners may prefer using an X-bar and R chart for continuous data, while others may use a p chart for proportion data.
2. Data selection: Different practitioners may choose different data sets for plotting control charts. For example, some practitioners may use a subgroup size of 2, while others may use a subgroup size of 5.
3. Control limits calculation: Control limits are calculated based on the data collected from the process and are used to determine if the process is in control or not. Different practitioners may use different methods to calculate the control limits, such as the classical method or the Exponentially Weighted Moving Average (EWMA) method.
4. Outlier detection: Different practitioners may have different criteria for identifying outliers in the data and removing them from the control chart.
5. Statistical software: Different practitioners may use different statistical software packages to set up control charts, which can result in different results and presentations of the data.

In summary, practitioner-to-practitioner variability in setting up a control chart can result from differences in the choice of chart type, data selection, control limits calculation, outlier detection, and statistical software.

Figure 19: ChatGPT’s imprecise explanation of the practitioner-to-practitioner variability.

Second, we suspect many within the SPC community had not considered using AI to generate SPC code before ChatGPT’s release. Our research background perhaps biases us, but we are excited about the idea of “pair programming” with a ChatGPT-like assistant. The advantages of “pair programming” (without the AI partner) are well-documented in computer science and software engineering (Begel and Nagappan, 2008). Our assessment, based on ChatGPT January 30 Version, captures the following: (a) ChatGPT seems to be more proficient in Python than R, which we suspect is based on the fact that OpenAI does not seem to have trained its code completion module in the R programming language (OpenAI, 2022b); (b) its understanding of the arguments in some of the R functions are limited; (c) ChatGPT impressed us with its ability to explain functions, including capturing that our inputted `ewma.ar10()` function is a Markov Chain-based approach; and (d) ChatGPT can be useful in translating code from one programming language to another, especially when the functions are widely-used (which can be helpful in deployment SPC methods in a production environment). That being said, any code written by ChatGPT would need to be validated. We do not think it will eliminate the SPC expertise required in code development; however, it can augment and reduce the time taken to develop, test, and deploy code.

Third, the current version of ChatGPT lacks the precision needed to correctly, consistently, and succinctly explain SPC concepts. In some instances, the generated output included the “correct terms,” but we judged the explanations to be of lower quality than expected from an SPC expert. In other instances, ChatGPT completely made-up explanations (e.g., in its description of the ‘bigfish’ dataset). This issue is a known limitation of current generative AI models, including ChatGPT (see Ji et al. (2022) for more details on *hallucination in natural language generation*). To be fair, we did

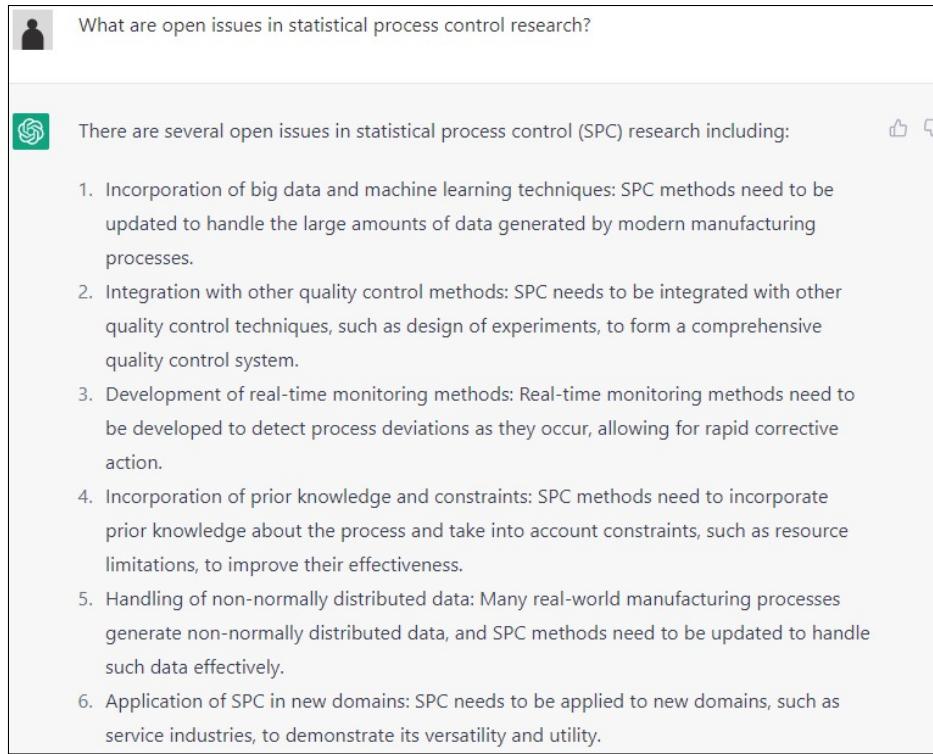


Figure 20: ChatGPT’s identification of six open issues in SPC research.

not attempt to optimize our prompts, especially in the practitioner and learning sections; however, our current prompts can be representative of what a practitioner and learner ask when they do not have additional context/details.

Fourth, the knowledge creation prompts and their associated outputs highlight that ChatGPT can be useful as an idea generation or brainstorming tool. For example, the topics mentioned in response to our syllabus and open areas for research were generally reasonable. However, the weight given to each listed topic can differ by individual (especially since there is no single correct answer here).

To supplement our discussion and potentially biased perspectives, we asked ChatGPT to provide us with its view of its limitations and why we should not use it in SPC. Its response (which we document in Our Supplementary Material for space limitations) included: a lack of domain/specialized knowledge of the SPC field as it is trained on a wide range of topics, potential bias from its training data, and no capacity for performing data analysis. In our estimation, these limitations support our overall recommendation that ChatGPT output needs to be validated by multiple sources and experts. In the case of code, the validation involves two levels: (a) does it work? (i.e., runs with no errors), and (b) does it work as expected? (i.e., to be validated against known answers). For non-code-based explanations, we are not in favor of using ChatGPT. However, we think its use for knowledge creation is appropriate if it is treated as an initial brainstorming tool.

## 8.2 Open Questions

Generative AI is a rapidly evolving field with many open questions. In our opinion, our SPC community can contribute to advancing the field by investigating the following questions:

- (a) How can we effectively evaluate the quality of the generated outputs? This question is not only limited to text-to-text-based LLMs, but also to models that output audio, images, and videos. Furthermore, existing approaches to evaluating the quality provide ad-hoc mechanisms to account for the output variability resulting from a singular prompt. For example, in the code translation model space, the “pass@k” metric is often used, where a model is said to pass if one out of the  $k$  generated output produces a correct translation (Weisz et al., 2023).
- (b) Poor sampling and training of generative AI models can lead to biased predictions. AI models can perpetuate and amplify biases when trained on biased data (Schwarz et al., 2021; Srinivasan and Uchino, 2021; Zhao

et al., 2018). In addition, generative AI models will collect and process possibly sensitive information, furthering our societal concerns on privacy and data security. We can see our community contributing in two complementary ways. First, our community can join/collaborate with generative AI companies to construct measures of training dataset bias and design sampling plans that eliminate/reduce the bias. In some domains, this is a relatively easy problem, e.g., in image generation, the gender distribution of the generated images can be used to detect dataset bias (Choi et al., 2020). However, in other application domains, this problem is much more challenging. The second approach involves designing experiments post the public release of a generative AI model to examine potential biases in its outputs.

- (c) Some release versions of generative AI models have APIs that allow for custom-built applications. For example, GPT-3 can be tailored (Lim et al., 2021) to SPC applications by providing it with SPC-based text for training (e.g., including Montgomery (2020) with relevant journal papers and code repositories). Similarly, ChatGPT can be trained by including text in the chat window (where long documents can be broken down into smaller chunks using existing open-source tools). It is yet to be determined whether such an approach can better answer SPC-specific questions and prompts.

In addition to these open questions, our community must be aware of many open ethical questions related to generative AI. These include new guidelines excluding LLM from scientific authorship (Editorial, 2023a; Nature, 2023; Thorp, 2023), discussion on ChatGPT’s impact on science (Editorial, 2023b; Stokel-Walker and Van Noorden, 2023), and how LLM outputs can be watermarked (Kirchenbauer et al., 2023) to prevent the misuse of AI-based output. We can also benefit from the examinations of LLMs’ use in other disciplines (Biswas, 2023; Dowling and Lucey, 2023; Gilson et al., 2023; Korinek, 2023).

### 8.3 Looking Forward

Several phenomena will shape the future of generative AI models. First, LLMs have doubled in size every six months (Sevilla et al., 2022). This has contributed to the rapid advancements in the abilities of these systems into usable application programming interfaces (APIs) that are widely used by the general public. Second, there is significant funding and investment interest in AI (see Constantz (2023) for an in-depth analysis of how that translated into recent earning calls). Third, for-profit companies have primarily pushed these generative AI models (as indicated in Figure 1a). Since the data used in training these models are unknown to the general public, there will likely be some ethical/legal ramifications if copyrights were infringed. Fourth, we suspect that industry-specific generative AI models will be made available, providing tailored for specific application areas. Fifth, the AI race between several big companies will likely change our online experience. For example, Microsoft announced that it is incorporating a version of ChatGPT in its “Bing” search toolbar/site and “Microsoft Edge” browser (see Mehdi (2023) for the news release).

The authors of this paper believe that generative AI and, specifically, LLMs will significantly change how we study, learn, and work. This is not the first technology to transform/disrupt society. Our society has transformed with many technological disruptions, including steam-powered machines, computers, the internet, and advances in wireless/cellular/sensing technologies. With these disruptions, statistical process control has evolved as well:

- (a) Univariate methods, such as the  $\bar{X}$  chart, were proposed to distinguish between common and special causes of variation. At the time, measurement was time-consuming, so the monitored statistic was univariate and sampled based on *rational subgroups*;
- (b) With some advancement in computational methods, multivariate charts were proposed to account for the correlation between several quality characteristics;
- (c) With the advancement in measurement and sensing technologies, it became possible to capture machine quality characteristics and a 100% sampling of manufacturing products geometries via image-based sensing. This led to the development of control charts that can be used in time-series (Alwan and Roberts, 1988), profile (Woodall, 2007) and functional data (Megahed et al., 2011; Wells et al., 2013) monitoring applications;
- (d) With the recent advancements in computational capabilities, our field has evolved to include statistical learning-based control charting methods (Weese et al., 2016), big data Megahed and Jones-Farmer (2015), and predictive monitoring strategies (Huberts et al., 2022).

In addition, our community’s productivity/efficiency has increased significantly with technological innovations in search engines, software (e.g., statistical analysis, version control, online meetings, and project management), and language models (which provide auto-complete suggestions in word processing, software, and browser applications).

At present, generative AI models like the one in ChatGPT are in their infancy. However, the future of these has the potential to revolutionize many fields, including SPC. Currently, ChatGPT can, at a minimum, be a tool to improve our efficiency and reduce the time needed to complete some tasks (e.g., templates for documents/emails, code translation/explanation, and brainstorming). In the future, we suspect generative AI tools will be incorporated into many software

tools. It is up to us as a community to examine how to use these innovations to further our practice, learning, and research agendas. Box and Woodall (2012, p. 20) stated that “quality and efficiency cannot compete against the right innovation ... [recounting] Deming’s story of the buggy whip manufacturer who had a highly efficient process with excellent quality but whose company collapsed because of the failure to foresee and adapt to the horseless carriage.” With generative AI likely to stay, we need to be proactive and creative in examining how to use this technology to add value to the SPC profession, and how the SPC profession can help advance this technology.

## Online Materials

Our GitHub Repository, [https://github.com/fmegahed/llm\\_expository](https://github.com/fmegahed/llm_expository), contains the code/data used in creating Figure 1, screenshots of all our ChatGPT interactions (under the `figs` subfolder), code we used to assess ChatGPT’s generated code for the Practice Section, and a Markdown containing videos of ChatGPT’s answers to each prompt in Figure 3. We provide these materials to document the outputs of our expository study given that the outputs from ChatGPT cannot be reproduced due to its inherent stochastic generation mechanism.

## References

- Agostinelli, A., T. I. Denk, Z. Borsos, J. Engel, M. Verzetti, A. Caillon, Q. Huang, A. Jansen, A. Roberts, M. Tagliasacchi, M. Sharifi, N. Zeghidour, and C. Frank (2023). MusicLM: generating music from text. arXiv article available at: <https://arxiv.org/abs/2301.11325>.
- Alwan, L. C. and H. V. Roberts (1988). Time-series modeling for statistical process control. *Journal of Business & Economic Statistics* 6(1), 87–95.
- Begel, A. and N. Nagappan (2008). Pair programming: what’s in it for me? In *Proceedings of the Second ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*, pp. 120–128.
- Biswas, S. (2023). ChatGPT and the future of medical writing. *Radiology*, 223312. InPress article available at: PMID: 36728748 and <https://doi.org/10.1148/radiol.223312>.
- Box, G. E. P. and W. H. Woodall (2012). Innovation, quality engineering, and statistics. *Quality Engineering* 24(1), 20–29.
- Brown, T., B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. (2020). Language models are few-shot learners. *Advances in Neural Information Processing Systems* 33, 1877–1901.
- Chakraborti, S., S. Human, and M. Graham (2008). Phase i statistical process control charts: An overview and some results. *Quality Engineering* 21(1), 52–62.
- Chen, M., J. Tworek, H. Jun, Q. Yuan, H. P. d. O. Pinto, et al. (2021). Evaluating large language models trained on code. arXiv article available at: <https://arxiv.org/abs/2107.03374>.
- Choi, K., A. Grover, T. Singh, R. Shu, and S. Ermon (2020, 13–18 Jul). Fair generative modeling via weak supervision. In H. D. III and A. Singh (Eds.), *Proceedings of the 37th International Conference on Machine Learning*, Volume 119 of *Proceedings of Machine Learning Research*, pp. 1887–1898. PMLR.
- Colosimo, B. M., E. del Castillo, L. A. Jones-Farmer, and K. Paynabar (2021). Artificial intelligence and statistics for quality technology: an introduction to the special issue. *Journal of Quality Technology* 53(5), 443–453.
- Constantz, J. (2023). California’s new gold rush: Big tech moves to gain the edge in AI. Bloomberg article available at: <https://www.bloomberg.com/news/articles/2023-02-03/big-tech-earnings-call-mentions-of-ai-spike-after-chatgpt-went-viral>.
- Crosier, R. B. (1986). A new two-sided cumulative quality control scheme. *Technometrics* 28(3), 187–194.
- Devlin, J., M.-W. Chang, K. Lee, and K. Toutanova (2018). BERT: pre-training of deep bidirectional transformers for language understanding. arXiv article available at: <https://arxiv.org/abs/1810.04805>.
- Dowling, M. and B. Lucey (2023). ChatGPT for (finance) research: The Bananarama conjecture. *Finance Research Letters*, 103662.

- Editorial (2023a). The AI writing on the wall. *Nature Machine Intelligence* 5(1). Available online at: <https://doi.org/10.1038/s42256-023-00613-9>.
- Editorial (2023b, January). Tools such as ChatGPT threaten transparent science; here are our ground rules for their use. *Nature*. Available online at: <http://dx.doi.org/10.1038/d41586-023-00191-1>.
- Gilson, A., C. W. Safranek, T. Huang, V. Socrates, L. Chi, R. A. Taylor, D. Chartash, et al. (2023). How does ChatGPT perform on the United States Medical Licensing Examination? the implications of large language models for medical education and knowledge assessment. *JMIR Medical Education* 9(1), e45312.
- Gozalo-Brizuela, R. and E. C. Garrido-Merchan (2023). ChatGPT is not all you need. a state of the art review of large generative AI models. arXiv article available at: <https://arxiv.org/abs/2301.04655>.
- Griffith, E. and C. Metz (2023, January). A new area of A.I. booms, even amid the tech gloom. Available at <https://www.nytimes.com/2023/01/07/technology/generative-ai-chatgpt-investments.html>. Last accessed on Jan 23, 2023.
- Hockman, K. K. and W. A. Jensen (2016). Statisticians as innovation leaders. *Quality Engineering* 28(2), 165–174.
- Huang, S., P. Grady, and GPT-3 (2022, September). Generative AI: A creative new world. Sequoia Capital. <https://www.sequoiacap.com/article/generative-ai-a-creative-new-world/>. Accessed: 2023-01-28.
- Huberts, L. C., M. Schoonhoven, and R. J. Does (2022). Multilevel process monitoring: A case study to predict student success or failure. *Journal of Quality Technology* 54(2), 127–143.
- Jardim, F. S., S. Chakraborti, and E. K. Epprecht (2020). Two perspectives for designing a phase II control chart with estimated parameters: The case of the Shewhart X-bar Chart. *Journal of Quality Technology* 52(2), 198–217.
- Ji, Z., N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y. Bang, A. Madotto, and P. Fung (2022). Survey of hallucination in natural language generation. *ACM Computing Surveys*. Available online at: <https://doi.org/10.1145/3571730>.
- Jones-Farmer, L. A., W. H. Woodall, S. H. Steiner, and C. W. Champ (2014). An overview of phase i analysis for process improvement and monitoring. *Journal of Quality Technology* 46(3), 265–280.
- Karpas, E., O. Abend, Y. Belinkov, B. Lenz, O. Lieber, N. Ratner, Y. Shoham, H. Bata, Y. Levine, K. Leyton-Brown, et al. (2022). Mrkl systems: A modular, neuro-symbolic architecture that combines large language models, external knowledge sources and discrete reasoning. arXiv article available at: <https://arxiv.org/abs/2205.00445>.
- Kim, K., M. A. Mahmoud, and W. H. Woodall (2003). On the monitoring of linear profiles. *Journal of Quality Technology* 35(3), 317–328.
- Kirchenbauer, J., J. Geiping, Y. Wen, J. Katz, I. Miers, and T. Goldstein (2023). A watermark for large language models. arXiv article available at: <https://arxiv.org/abs/2301.10226>.
- Knoth, S. (2021). Steady-state average run length(s) – methodology, formulas and numerics. *Sequential Analysis* 40(3), 405–426.
- Korinek, A. (2023). Language models and cognitive automation for economic research. NBER Working Paper available at: <https://www.nber.org/papers/w30957>.
- Lim, R., M. Wu, and L. Miller (2021, December). Customizing GPT-3 for your application. OpenAI. <https://openai.com/blog/customized-gpt-3/>. Last accessed on Feb 10, 2023.
- Lucas, J. M. and M. S. Saccucci (1990). Exponentially weighted moving average control schemes: properties and enhancements. *Technometrics* 32(1), 1–12.
- Maleki, M. R., A. Amiri, and P. Castagliola (2018). An overview on recent profile monitoring papers (2008–2018) based on conceptual classification scheme. *Computers & Industrial Engineering* 126, 705–728.
- Markov, A. A. (2006). An example of statistical investigation of the text eugene onegin concerning the connection of samples in chains. *Science in Context* 19(4), 591–600.
- McKee, F. and D. Noever (2022). Chatbots in a botnet world. arXiv article available at: <https://arxiv.org/abs/2212.11126>.

- Megahed, F. M. (2019). Discussion on “real-time monitoring of events applied to syndromic surveillance”. *Quality Engineering* 31(1), 97–104.
- Megahed, F. M. and L. A. Jones-Farmer (2015). Statistical perspectives on “big data”. In S. Knoth and W. Schmid (Eds.), *Frontiers in Statistical Quality Control* 11, pp. 29–47. Springer.
- Megahed, F. M., W. H. Woodall, and J. A. Camelio (2011). A review and perspective on control charting with image data. *Journal of Quality Technology* 43(2), 83–98.
- Mehdi, Y. (2023). Reinventing search with a new AI-powered Microsoft Bing and Edge, your copilot for the web. Microsoft Blog available at <https://blogs.microsoft.com/blog/2023/02/07/reinventing-search-with-a-new-ai-powered-microsoft-bing-and-edge-your-copilot-for-the-web/>.
- Michel, J.-B. and E. L. Aiden (2011). What we learned from 5 million books. TEDxBoston. The TEDx talk is available at [https://www.ted.com/talks/jean\\_baptiste\\_michel\\_erez\\_lieberman\\_aiden\\_what\\_we\\_learned\\_from\\_5\\_million\\_books](https://www.ted.com/talks/jean_baptiste_michel_erez_lieberman_aiden_what_we_learned_from_5_million_books).
- Montgomery, D. (2020). *Introduction to Statistical Quality Control* (8 ed.). John Wiley & Sons. ISBN: 9781119723097.
- Nature (2023). For authors: Initial submission guidelines. <https://www.nature.com/nature/for-authors/initial-submission>. Last accessed on February 12, 2023.
- Noorossana, R., A. Saghaei, and A. Amiri (2013). *Statistical analysis of profile monitoring* (1 ed.). Wiley Online Library. ISBN: 9780470903223.
- OpenAI (2022a, November). ChatGPT: Optimizing language models for dialogue. Available at: <https://openai.com/blog/chatgpt/>. Last accessed on January 28, 2023.
- OpenAI (2022b, November). Code completion. Available at: <https://platform.openai.com/docs/guides/code/introduction>. Last accessed on February 09, 2023.
- OpenAI (2022c, April). DALL.E 2. Available at: <https://openai.com/dall-e-2/>. Last accessed on January 27, 2023.
- Ouyang, L., J. Wu, X. Jiang, D. Almeida, C. L. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, et al. (2022). Training language models to follow instructions with human feedback. arXiv article available at: <https://arxiv.org/abs/2203.02155>.
- Radford, A., K. Narasimhan, T. Salimans, I. Sutskever, et al. (2018). Improving language understanding by generative pre-training. OpenAI Preprint available at: [https://cdn.openai.com/research-covers/language-unsupervised/language\\_understanding\\_paper.pdf](https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf).
- Saleh, N. A., M. A. Mahmoud, M. J. Keefe, and W. H. Woodall (2015). The difficulty in designing Shewhart X and X-bar control charts with estimated parameters. *Journal of Quality Technology* 47(2), 127–138.
- Schick, T., J. Dwivedi-Yu, R. Dessì, R. Raileanu, M. Lomeli, L. Zettlemoyer, N. Cancedda, and T. Scialom (2023). Toolformer: Language models can teach themselves to use tools. arXiv article available at: <https://arxiv.org/abs/2302.04761>.
- Schwarz, K., Y. Liao, and A. Geiger (2021). On the frequency bias of generative models. *Advances in Neural Information Processing Systems* 34, 18126–18136.
- Scrucca, L. (2004). qcc: an R package for quality control charting and statistical process control. *R News*. Vol. 4/1. July 8, 2004. Available online at: [http://www.stat.unipg.it/lucamisc/Rnews\\_2004-1-pag11-17.pdf](http://www.stat.unipg.it/lucamisc/Rnews_2004-1-pag11-17.pdf).
- Scrucca, L. (2017, July). A quick tour of qcc. [https://cran.r-project.org/web/packages/qcc/vignettes/qcc\\_a\\_quick\\_tour.html](https://cran.r-project.org/web/packages/qcc/vignettes/qcc_a_quick_tour.html). Accessed: 2023-02-02.
- Sevilla, J., L. Heim, A. Ho, T. Besiroglu, M. Hobbahn, and P. Villalobos (2022). Compute trends across three eras of machine learning. arXiv article available at: <https://arxiv.org/abs/2202.05924>.
- Shannon, C. E. (1948). A mathematical theory of communication. *The Bell System Technical Journal* 27(3), 379–423.

- Singer, U., A. Polyak, T. Hayes, X. Yin, J. An, S. Zhang, Q. Hu, H. Yang, O. Ashual, O. Gafni, D. Parikh, S. Gupta, and Y. Taigman (2022). Make-A-Video: text-to-video generation without text-video data. arXiv article available at: <https://arxiv.org/abs/2209.14792>.
- Singer, U., S. Sheynin, A. Polyak, O. Ashual, I. Makarov, F. Kokkinos, N. Goyal, A. Vedaldi, D. Parikh, J. Johnson, and Y. Taigman (2023). Text-To-4D dynamic scene generation. arXiv article available at: <https://arxiv.org/abs/2301.11280>.
- Srinivasan, R. and K. Uchino (2021). Biases in generative art: A causal look from the lens of art history. In Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency, pp. 41–51.
- Stokel-Walker, C. and R. Van Noorden (2023). What ChatGPT and generative AI mean for science. Nature 614(7947), 214–216.
- Thorp, H. H. (2023). ChatGPT is fun, but not an author. Science 379(6630), 313–313.
- Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin (2017). Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.), Advances in Neural Information Processing Systems, Volume 30. Curran Associates, Inc. Paper available at: <https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fb053c1c4a845aa-Paper.pdf>.
- Weese, M., W. Martinez, F. M. Megahed, and L. A. Jones-Farmer (2016). Statistical learning methods applied to process monitoring: An overview and perspective. Journal of Quality Technology 48(1), 4–24.
- Wei, J. and Y. Tay (2022, November). Characterizing emergent phenomena in large language models. Google Research Blog available at: <https://ai.googleblog.com/2022/11/characterizing-emergent-phenomena-in.html>. Last accessed on Jan 27, 2023.
- Wei, J., Y. Tay, R. Bommasani, C. Raffel, B. Zoph, S. Borgeaud, D. Yogatama, M. Bosma, D. Zhou, D. Metzler, E. H. Chi, T. Hashimoto, O. Vinyals, P. Liang, J. Dean, and W. Fedus (2022). Emergent abilities of large language models. Transactions on Machine Learning Research. Accepted paper, available at <https://openreview.net/forum?id=yzkSU5zdwD>.
- Weisz, J. D., M. Muller, J. He, and S. Houde (2023). Toward general design principles for generative AI applications. arXiv article available at: <https://arxiv.org/abs/2301.05578>.
- Wells, L. J., F. M. Megahed, C. B. Niziolek, J. A. Camelio, and W. H. Woodall (2013). Statistical process monitoring approach for high-density point clouds. Journal of Intelligent Manufacturing 24, 1267–1279.
- Woodall, W. H. (2007). Current research on profile monitoring. Production 17, 420–425.
- Zhang, M., Y. Peng, A. Schuh, F. M. Megahed, and W. H. Woodall (2013). Geometric charts with estimated control limits. Quality and Reliability Engineering International 29(2), 209–223.
- Zhao, S., H. Ren, A. Yuan, J. Song, N. Goodman, and S. Ermon (2018). Bias and generalization in deep generative models: An empirical study. Advances in Neural Information Processing Systems 31.
- Zwetsloot, I. M., L. A. Jones-Farmer, and W. H. Woodall (2023). Monitoring univariate processes using control charts: Some practical issues and advice. Submitted for Publication.