

The use of Copilot, a generative artificial intelligence tool, as VBA programming assistant in Excel-based Health Economics models

MSR26

View all of Parexel's posters at ISPOR Europe



J.E. Poirrier ¹, R. Bergemann ²
¹ Parexel International, Wavre, Belgium ² Parexel International, Basel, Switzerland

Background

Copilot is a Generative Artificial Intelligence (AI) tools launched in 2022 by GitHub / Microsoft (MS) [1]. Based on a codex from Open Source repositories on GitHub and Open AI GPT-3 [2], Copilot can provide AI-based coding suggestions that match a project's context and style conventions.

Objectives

Following dithyrambic reviews [3], we wanted 1) to test how health economists can take advantage of Copilot in their cost-effectiveness model developments in MS-Excel / Visual Basic for Applications (VBA) and 2) to draw attention on some non-programming issues related to the use of AI in programming for health economics.

Methods

An existing, MS-Excel based cost-effectiveness model (CEM) for the treatment of Urinary Tract Infection in hospitalized patients was stripped from its VBA code 1) for some user-interface (UI) interactions, 2) for the creation of the cost-effectiveness (CE) frontier and 3) for the Probabilistic Sensitivity Analysis

(PSA).
Copilot was then prompted to rewrite code from scratch, first with general prompts, then with step-by-step requests. The generated code was analyzed by an experienced VBA developer to assess quality,

verboseness, and performance.
Software versions: MS-Excel 365 2310, Visual Studio Code x64 1.81.1, GitHub Copilot Extension 1.129.0, xlwings 0.30.12, VBA Extension 0.9.3.

Results

Copilot to create user-interface interactions

- Creating a subroutine, interacting with a (named) custom ribbon and calling the appropriate sheet is easy for Copilot, given the appropriate prompt (Figure 1). This uses standard Excel/VBA syntax.
- Given the task to call a user form (for setting a sheet zoom level), Copilot struggled to perform introspection of the form's custom properties and values. It however learnt from other parts of the code how to re-use code to re-position the form, going even to copy comments from another part of the model (Figure 2).
- Given the task to hide or show comparators rows based on a switch, Copilot could read the Boolean value from the designated cell but failed at selecting the right rows to hide or show. Indeed, Copilot can't read the Excel file (just the VBA code). Therefore, it can't know where comparators are located.

Copilot to create the CE frontier

- Prompted with just "create the cost-effectiveness frontier", Copilot couldn't do anything since this concept isn't (yet) in its codex.
- A programmer could describe to Copilot the algorithm, line by line, and it would generate the appropriate code. Exact cells or ranges had to be given to perform the right calculations.
- Letting Copilot generate the CE frontier (designated as "scatter plot") gives a surprisingly similar graph (Figure 3), with some minor issues.

General considerations on the use of Copilot in Health Economics models

- Copilot depends heavily on a pre-defined codex to present suggestions. VBA and health economics are underrepresented compared to general data analytics and languages like R or Python.
- Given the very detailed prompts compensating for the lack of Copilot expertise in VBA, the validity and reproducibility of Copilot suggestions couldn't be verified here. Explainability, validity and

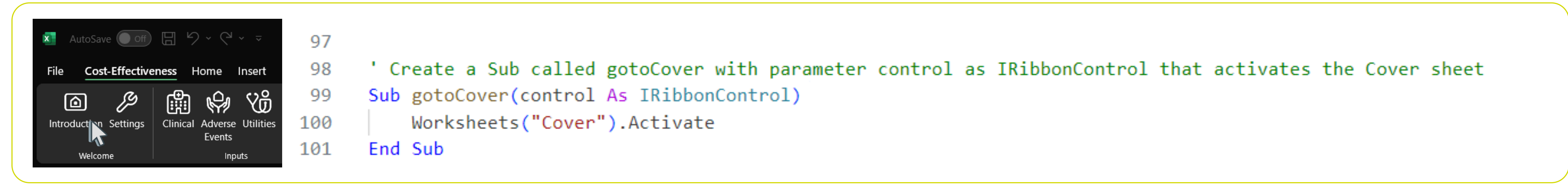


Figure 1. Part of a custom ribbon and Copilot-generated VBA code to activate a specific sheet

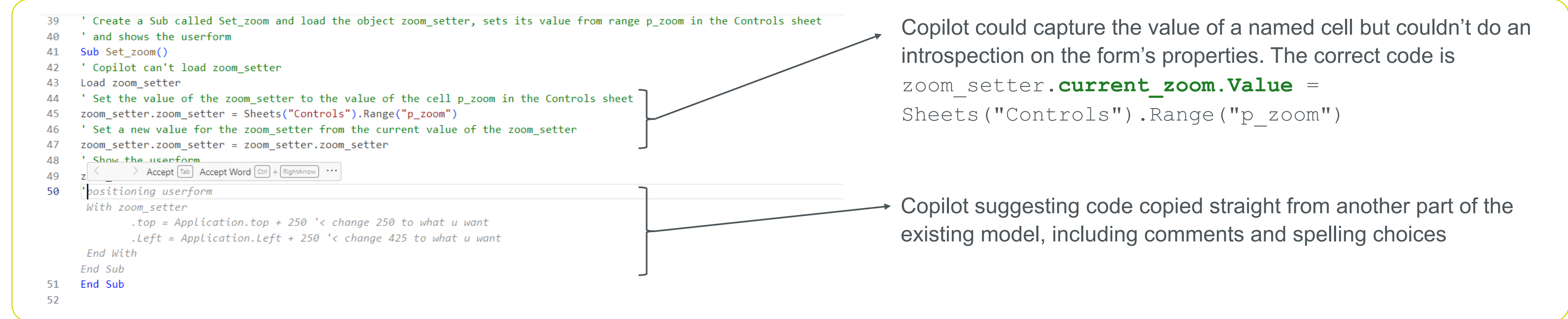


Figure 2. Copilot-generated VBA code to load a custom form, set a zoom level and repositioning the form

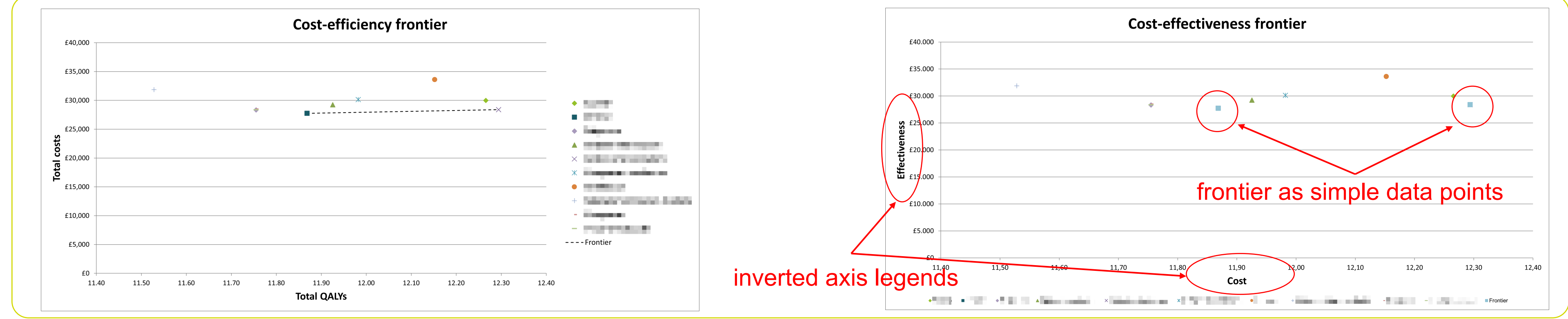


Figure 3. Original CE plane (left) and Copilot-generated CE plane (right, issues highlighted in red)

Copilot to create a PSA

Similar challenges arose:

- Copilot doesn't know how to create a PSA in VBA from a general prompt.
- Copilot should be guided very precisely on which cells and ranges to read from and to write to as it misses the context of the sheets' content.

- Copilot could easily generate a scatterplot, given a set of (x, y) pair of data but didn't figure out exactly some of the nuances (e.g., 95% confidence ellipse, WTP at different thresholds).
- Since guided step-by-step, the performance of Copilot code was similar to a human code.

Conclusions

- Copilot can generate VBA code in a CEM.
- The quality and verboseness of the generated code depends on the user's detailed request. In addition,
 - Simple UI-altering code can easily be generated
 - Health Economics-specific code needs to be prompted step-by-step
- Copilot could accelerate model development for junior developers
- Experienced programmers might wait for improvements from GPT-4 and an increased VBA / health economics codex.
- Issues in code explainability, ownership and licensing need to be clarified.

REFERENCES

[1] Dohmke, T. **GitHub Copilot is generally available to all developers**, 2022: <https://github.blog/2022-06-21-github-copilot-is-generally-available-to-all-developers/>
[2] Wikipedia. **GPT-3**, 2023: <https://en.wikipedia.org/wiki/GPT-3>
[3] Dohmke, T., M. Iansiti, and G. Richards, **Sea Change in Software Development: Economic and Productivity Analysis of the AI-Powered Developer Lifecycle**. arXiv preprint arXiv:2306.15033, 2023
[4] Busch, K.E., **Generative Artificial Intelligence and Data Privacy: A Primer**, 2023, Congressional Research Service, R47569
[5] European Commission, **On Artificial Intelligence - A European approach to excellence and trust**, 2020, COM(2020) 65 final
[6] Moradi Dakhel, A., et al., **GitHub Copilot AI pair programmer: Asset or Liability?** Journal of Systems and Software, 2023. 203: p. 111734.

