helpful. To incorporate human feedback with `AutoGen`, one can set `human_input_mode='ALWAYS'` in the user proxy agent. We select one challenging problem that none of these systems can solve autonomously across three trials. We adhere to the process outlined below to provide human inputs for all the compared methods:

1. Input the problem: `Find the equation of the plane which bisects the angle between the planes` $3x - 6y + 2z + 5 = 0$ `and` $4x - 12y + 3z - 3 = 0$, `and which contains the point` $(-5, -1, -5)$. `Enter your answer in the form`

$$Ax + By + Cz + D = 0,$$

   `where` A, B, C, D `are integers such that` $A > 0$ `and` $\gcd(|A|, |B|, |C|, |D|) = 1$.
2. The response from the system does not solve the problem correctly. We then give a hint to the model: `Your idea is not correct. Let's solve this together. Suppose` $P = (x, y, z)$ `is a point that lies on a plane that bisects the angle, the distance from P to the two planes is the same. Please set up this equation first.`
3. We expect the system to give the correct distance equation. Since the equation involves an absolute sign that is hard to solve, we would give the next hint: `Consider the two cases to remove the abs sign and get two possible solutions.`
4. If the system returns the two possible solutions and doesn't continue to the next step, we give the last hint: `Use point (-5,-1,-5) to determine which is correct and give the final answer.`
5. Final answer is $\boxed{\text{11x+6y+5z+86=0}}$.

We observed that `AutoGen` consistently solved the problem across all three trials. ChatGPT+Code Interpreter and ChatGPT+Plugin managed to solve the problem in two out of three trials, while AutoGPT failed to solve it in all three attempts. In its unsuccessful attempt, ChatGPT+Code Interpreter failed to adhere to human hints. In its failed trial, ChatGPT+Plugin produced an almost correct solution but had a sign discrepancy in the final answer. AutoGPT was unable to yield a correct solution in any of the trials. In one trial, it derived an incorrect distance equation. In the other two trials, the final answer was incorrect due to code execution errors.

Scenario 3: Multi-User Problem Solving. Next-generation LLM applications may necessitate the involvement of multiple real users for collectively solving a problem with the assistance of LLMs. We showcase how `AutoGen` can be leveraged to effortlessly construct such a system. Specifically, building upon scenario 2 mentioned above, we aim to devise a simple system involving two human users: a student and an expert. In this setup, the student interacts with an LLM assistant to address some problems, and the LLM automatically resorts to the expert when necessary.

The overall workflow is as follows: The student chats with the LLM-based assistant agent through a student proxy agent to solve problems. When the assistant cannot solve the problem satisfactorily, or the solution does not match the expectation of the student, it would automatically hold the conversation and call the pre-defined `ask_for_expert` function via the *function_call* feature of GPT in order to resort to the expert. Specifically, it would automatically produce the initial message for the `ask_for_expert` function, which could be the statement of the problem or the request to verify the solution to a problem, and the expert is supposed to respond to this message with the help of the expert assistant. After the conversation between the expert and the expert's assistant, the final message would be sent back to the student assistant as the response to the initial message. Then, the student assistant would resume the conversation with the student using the response from the expert for a better solution. A detailed visualization is shown in Figure 6.

With `AutoGen`, constructing the student/expert proxy agent and the assistant agents is straightforward by reusing the built-in `UserProxyAgent` and `AssistantAgent` through appropriate configurations. The only development required involves writing several lines of code for the `ask_for_expert` function, which then becomes part of the configuration for the assistant. Additionally, it's easy to extend such a system to include more than one expert, with a specific `ask_for_expert` function for each, or to include multiple student users with a shared expert for consultation.