A. Additional Baselines and Ablation Studies

Beyond the evaluation in Sec. 5, we test additional baselines, including reasoning models that have proven successful in program synthesis tasks [89]. We evaluate QWEN$_{2.5}$ CODER (14B) [75], which complements QWEN$_{2.5}$ CODER (32B) from Sec. 5.2, and reasoning models from the DEEPSEEK-R1 QWEN family (14B and 32B) [90]. We also evaluate TIKZERO (Base), a variant of TIKZERO (Cos) without the trainable probe and gating mechanism, to assess their contributions.

As shown in Tab. 5, TIKZERO (Cos) achieves the highest performance, surpassing TIKZERO (Base) on both DREAMSIM and CLIPSCORE metrics and in average performance. These results validate the probe and gate design. Additionally, QWEN$_{2.5}$ CODER (14B) performs worse than both TIKZERO (Cos) and, as expected, its 32B variant in Tab. 3. The results are consistent with our findings in Sec. 5.1 that TIKZERO (Cos) outperforms end-to-end trained baselines of comparable size. Notably, the reasoning models show the lowest overall performance, even compared to QWEN$_{2.5}$ CODER (14B), indicating that reasoning capabilities alone are insufficient for graphics program synthesis and more domain-specific post-training may be needed.

B. Supplementary Comparison with DETIKZIFY

Tab. 6 shows in detail how TIKZERO's inverse graphics model (hereafter referred to as DETIKZIFY$_{v2}$) compares against DETIKZIFY$_{DS}$ (7b), previously the best performing DETIKZIFY model, as evaluated on the test split of DATIKZ$_{v3}$. DETIKZIFY$_{v2}$ clearly outperforms its predecessor across all evaluated metrics. Below, we briefly outline key differences in training and inference beyond what we described in Sec. 4. For a comprehensive description of the foundation on which DETIKZIFY$_{v2}$ builds, we refer to Belouadi et al. [2].

Training Similar to DETIKZIFY, DETIKZIFY$_{v2}$ employs a dense layer as the modality connector between the vision encoder and text decoder. However, for pretraining this layer, we replace the METAFIG dataset [2] with the substantially larger ARXIVCAP dataset, extracting 1 million (figure, caption, OCR) triplets. During fine-tuning, we randomly substitute inputs with synthetically generated sketches to support hand-drawn inputs. To generate these sketches, we fine-tune the image-editing model ULTRAEDIT [91] on a dataset of real, human-created scientific sketches [2]. The resulting model, ULTRASKETCH, achieves a congruence coefficient (CC) [92] of 0.74 with said sketches, compared to 0.72 for the previous model used with DETIKZIFY. Additionally, we generate synthetic sketches using traditional image transformations such as random displacement fields. While these sketches exhibit less diversity, they better preserve text rendering and achieve a comparable CC of 0.75. Averaging the sketch representations from both methods increases the CC to 0.82, demonstrating their complementary nature.

Inference DETIKZIFY implements a Monte Carlo Tree Search-based inference algorithm to iteratively refine outputs. As a reward signal r, it computes the cosine similarity $r_{\cos} = \cos(\text{pool}(\boldsymbol{x}), \text{pool}(\boldsymbol{y}))$ between image patch embeddings $\boldsymbol{x}, \boldsymbol{y}$ of input images and compiled outputs via a learned pooling function. Since DETIKZIFY$_{v2}$ fully fine-tunes the vision encoder and uses its patch embeddings directly, it cannot compute pooled embeddings in the same way. As an alternative, inspired by popular machine translation metrics [93–96], we experiment with computing the Earth Mover's Distance (EMD) [97, 98] with image patch embeddings. Given the distance matrix $\boldsymbol{D}$, where $D_{i,j} = \cos(x_i, y_j)$, EMD is defined as follows:

$$
\begin{aligned}
\text{EMD}(\boldsymbol{x}, \boldsymbol{y}) &= \frac{\sum_{i=1}^{|\boldsymbol{x}|} \sum_{j=1}^{|\boldsymbol{y}|} F_{i,j} D_{i,j}}{\sum_{i=1}^{|\boldsymbol{x}|} \sum_{j=1}^{|\boldsymbol{y}|} F_{i,j}}, \\
\text{with} \quad & \min_{\boldsymbol{F} \geq 0} \sum_{i=1}^{|\boldsymbol{x}|} \sum_{j=1}^{|\boldsymbol{y}|} F_{i,j} D_{i,j} \\
\text{s.t.} \quad & \forall_{i,j} \begin{cases} \sum_{i=1}^{|\boldsymbol{x}|} F_{i,j} = \frac{1}{|\boldsymbol{y}|}, \\ \sum_{j=1}^{|\boldsymbol{y}|} F_{i,j} = \frac{1}{|\boldsymbol{x}|}. \end{cases}
\end{aligned} \tag{2}
$$

When correlating reward scores computed as $r_{\cos}$ from DETIKZIFY and $r_{\text{EMD}} = \text{EMD}(x_i, y_j)$ from DETIKZIFY$_{v2}$ with human judgments from Belouadi et al. [2], we find that r_{EMD} enhances correlation with humans (0.456 segment-level and 0.911 system-level Spearman's ρ), compared to $r_{\cos}$ (0.436 and 0.642, respectively). This demonstrates that DETIKZIFY$_{v2}$ not only supports the inference algorithm but improves upon DETIKZIFY's capabilities.

C. Supplementary Inference Details

To instruct general-purpose models to generate Ti*k*Z code, we employ a consistent prompt across all models (GPT-4o, QWEN$_{2.5}$ CODER (32B), and IDEFICS 3 (8B)) originally engineered by Zhang et al. [4]. For each figure, we replace the `<caption>` placeholder with the specific caption:

```
Please generate a scientific figure according to the following requirements: <caption>. Your output should be in TikZ code. Do not include any text other than the TikZ code.
```

D. Supplementary Experimental Results

Tab. 7 presents detailed evaluation metrics scores for the low-resource training experiments discussed in Sec. 6.2. The results show a consistent degradation in performance across all metrics as both the amount of training data and the number of layers decrease, a trend effectively captured by the AVG scores also shown in Tab. 4.

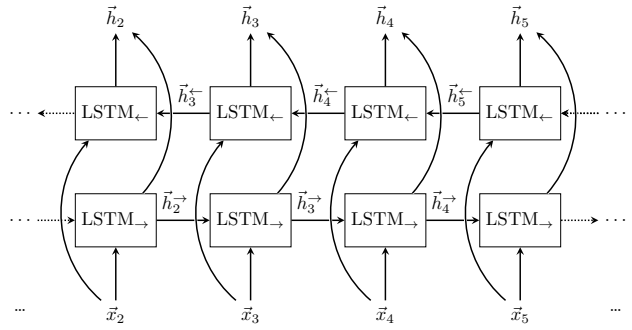
E. Annotator Demographics

Our annotation team consists of thirteen experts with extensive research experience in Machine Learning, Natural Language Processing, or Computer Vision. The team includes one male faculty member, four female PhD students, four male PhD students, and four male researcher scientists from a research institute. We deliberately selected expert annotators based on findings by Belouadi et al. [3], which demonstrated that crowd workers often lack the necessary research background to provide reliable annotations for scientific figures. To mitigate potential biases, each annotator received the tuples and items within the tuples in randomized order.

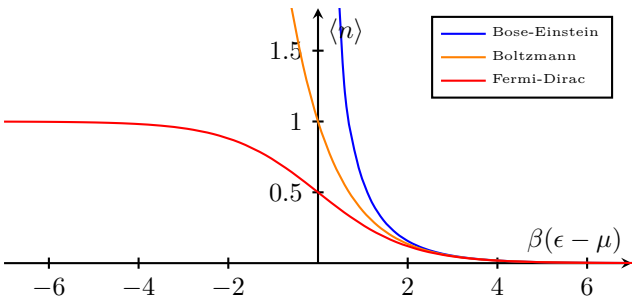
F. Additional Examples

Figure 5 showcases examples[5] from DaTi*k*Z$_{v3}$ with permissive licenses. Additionally, Tab. 8 presents randomly sampled tuples from our human evaluation with the highest and lowest rated instances highlighted. The results show that AutomaTi*k*Z$_{v2}$ (LLM) and Ti*k*Zero (Cos) are more frequently selected as the worst models (four and three times, respectively), while Ti*k*Zero+ and GPT-4o are more often chosen as the best models (both three times), which aligns with our findings in Sec. 5.3. Finally, Fig. 6 illustrates example programs generated by Ti*k*Zero+ and AutomaTi*k*Z$_{v2}$ (LLM), demonstrating how Ti*k*Zero+ utilizes advanced Ti*k*Z features, whereas AutomaTi*k*Z$_{v2}$ (LLM) employs only basic, simple commands.

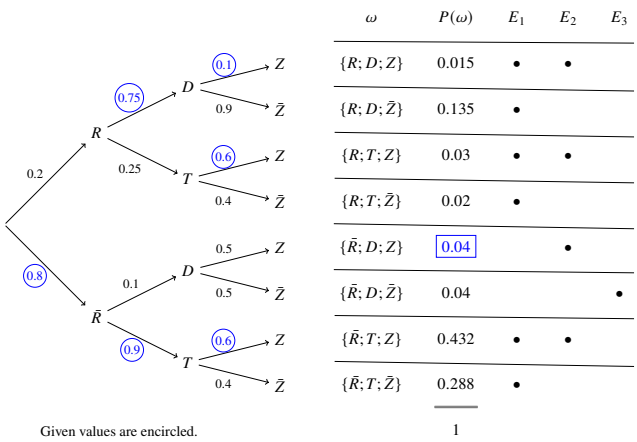
[5] sourced from https://github.com/PetarV-/TikZ, https://github.com/janosh/tikz, https://tikz.net, and https://arxiv.org



(a) A diagram representing a recurrent neural network consisting of several LSTM blocks, processing the input sequence simultaneously forwards and backwards (to exploit both directions of temporal dependence). Contains some rather tight manoeuvering.



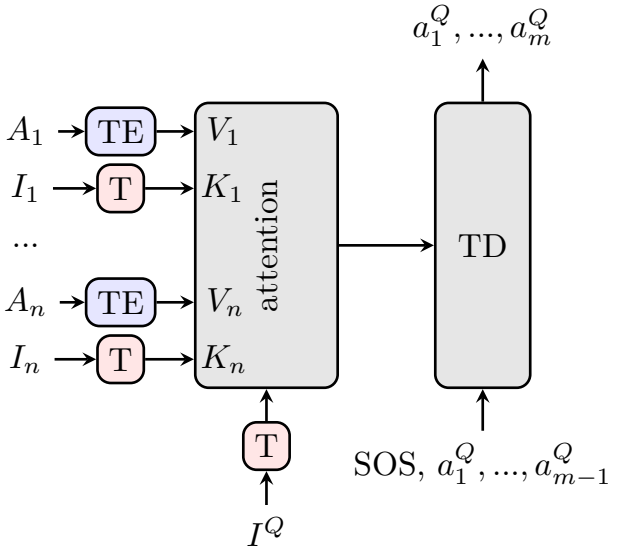
(b) A plot comparing the distribution functions of Bose-Einstein, Boltzmann, and Fermi-Dirac statistics as a function of the reduced chemical potential $\beta(\epsilon - \mu)$. This visualiation highlights the differences between the three types of distribution functions, which are used to describe the behavior of particles in different statistical systems.



ω	$P(\omega)$	E_1	E_2	E_3
$\{R;D;Z\}$	0.015	•	•	
$\{R;D;\bar{Z}\}$	0.135	•		
$\{R;T;Z\}$	0.03	•	•	
$\{R;T;\bar{Z}\}$	0.02	•		
$\{\bar{R};D;Z\}$	0.04		•	
$\{\bar{R};D;\bar{Z}\}$	0.04			•
$\{\bar{R};T;Z\}$	0.432	•	•	
$\{\bar{R};T;\bar{Z}\}$	0.288	•		
	1			

Given values are encircled.

(c) Tree with aligned matrix. A probability tree with an aligned matrix listing the possible outcomes, their probabilities and three columns for events described in later tasks. It uses the grahdrawing library and requires LuaLaTeX.



(d) Our approach is a modified version of **meta-seq2seq**. A transformer decoder (TD) is trained to produce a sequence of actions $a_1^Q, \ldots, a_m^Q$ given a query instruction I^Q. The context are demonstrations (I_k, A_k) produced by our generative model. We use a transformer encoder-decoder (T) to encode instructions and state S and a transformer encoder (TE) to encode actions. The transformers that process instructions (pink blocks) receive state S as the input of the encoder.

Figure 5. Representative examples from DaTikZ$_{v3}$ (also present in DaTikZ and DaTikZ$_{v2}$), with permissive licenses.

Models	**DSim**$\uparrow$	**KID**$\downarrow$	**CLIP**$\uparrow$	**cBLEU**$\uparrow$	**TED**$\downarrow$	**MTE**$\uparrow$	**AVG**$\uparrow$
TikZero (Cos)	**52.829**	**5.103**	10.051	**1.603**	65.51	82.291	**64.309**
TikZero (Base)	52.373	5.225	9.428	1.589	65.286	83.128	63.129
Qwen$_{2.5}$ Coder (14B)	48.352	12.988	19.761	0.229	**60.304**	**93.285**	58.894
DeepSeek-R1 Qwen (32B)	47.573	8.887	21.201	1.388	64.928	66.225	57.252
DeepSeek-R1 Qwen (14B)	44.616	15.43	**21.695**	0.842	63.323	36.11	31.102

Table 5. System-level scores × 100 for TikZero (Cos) and additional baselines. Overall, TikZero achieves the strongest average performance across metrics.

Models	Reference Figures					Synthetic Sketches				
	DSim$_\uparrow$	KID$_\downarrow$	cBLEU$_\uparrow$	TED$_\downarrow$	MTE$_\uparrow$	DSim$_\uparrow$	KID$_\downarrow$	cBLEU$_\uparrow$	TED$_\downarrow$	MTE$_\uparrow$
DeTikZify$_{DS}$ (7b)	75.46	0.842	2.953	56.851	84.019	67.379	0.766	1.541	59.589	84.401
DeTikZify$_{v2}$	**80.503**	**0.626**	**6.105**	**54.946**	**93.326**	**74.584**	**0.751**	**3.356**	**58.32**	**93.858**

Table 6. System-level scores × 100 for DeTikZify$_{v2}$ and DeTikZify$_{DS}$ (7b) on both reference figures and synthetic sketches generated with UltraSketch from the test split of DaTikZ$_{v3}$. Best scores are in bold, and arrows indicate metric directionality. Note that we compute DreamSim using updated models [68], whereas Belouadi et al. [3] used the original models in their work [67].

Data	Intv.	DSim$_\uparrow$	KID$_\downarrow$	CLIP$_\uparrow$	cBLEU$_\uparrow$	TED$_\downarrow$	MTE$_\uparrow$	AVG$_\uparrow$
100%	1	**52.771**	5.127	9.949	**1.607**	65.516	82.292	**92.411**
100%	2	52.311	5.2	**9.955**	1.484	65.473	82.588	87.557
100%	4	51.794	5.688	8.886	1.429	65.399	**83.988**	82.254
100%	8	51.59	5.933	9.818	1.371	65.608	83.679	76.545
50%	1	52.106	5.835	8.527	1.454	65.605	83.599	77.478
50%	2	52.143	**5.103**	9.315	1.393	**65.355**	82.924	85.249
50%	4	50.492	6.689	8.852	1.459	65.951	78.456	47.381
50%	8	50.093	6.738	7.999	1.379	65.963	78.923	40.816
25%	1	51.55	6.055	9.12	1.472	66.237	77.961	49.967
25%	2	51.231	6.152	8.943	1.43	65.714	77.566	54.942
25%	4	49.859	7.715	7.316	1.41	66.128	79.704	32.12
25%	8	49.179	7.764	6.495	1.434	66.009	79.9	29.774
12.5%	1	50.485	6.25	7.568	1.509	65.8	80.816	56.055
12.5%	2	50.152	7.129	6.353	1.275	66.045	81.05	33.817
12.5%	4	49.667	7.031	6.474	1.221	65.892	82.634	37.914
12.5%	8	48.827	8.154	5.054	1.11	65.813	80.738	16.25

Table 7. System-level scores × 100 TikZero (Cos) trained on varying fractions of data and intervals of cross-attention layers. Bold and underlined values denote the best and second-best scores for the whole table, respectively. Cell shading illustrates score magnitudes. Arrows indicate metric directionality.

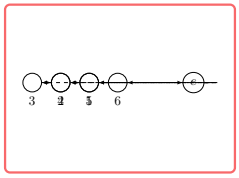
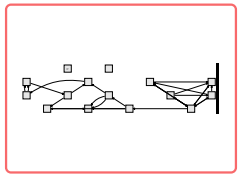
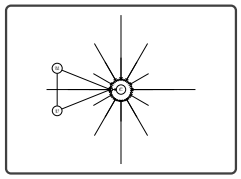
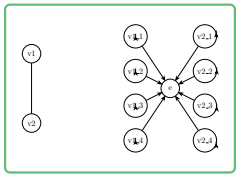
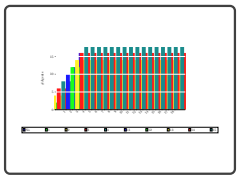
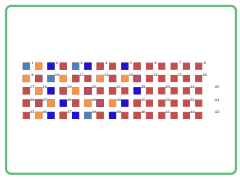
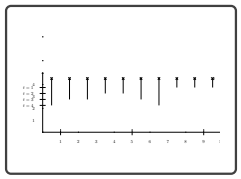
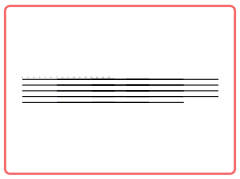
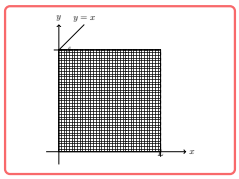
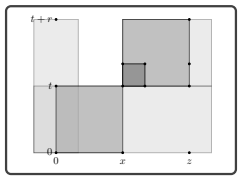
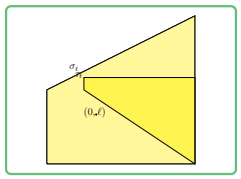
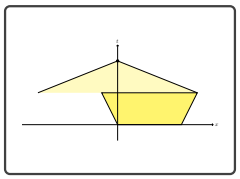
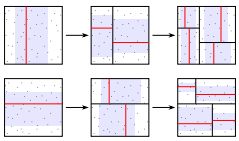
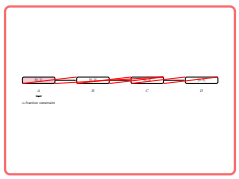
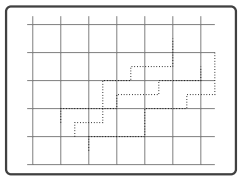
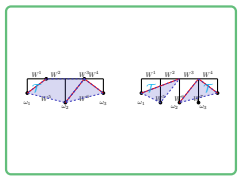
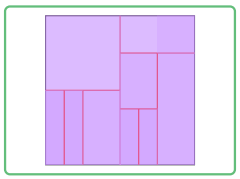
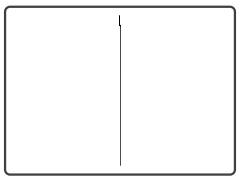
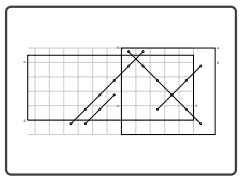
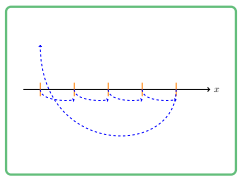
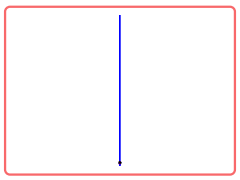
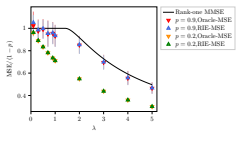
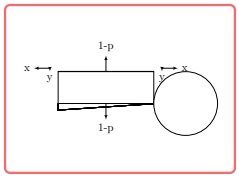
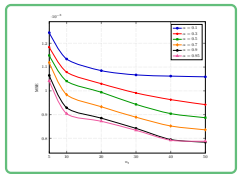
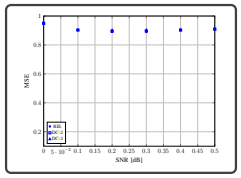
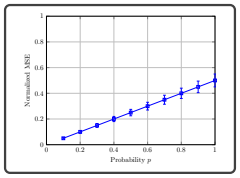
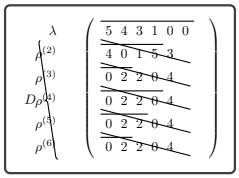
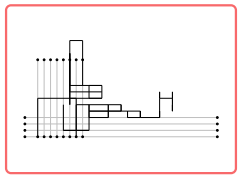
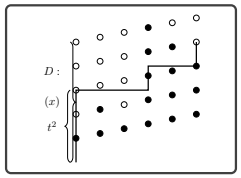
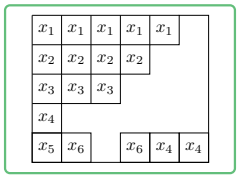
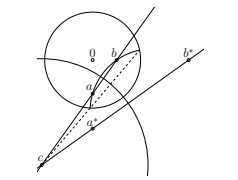
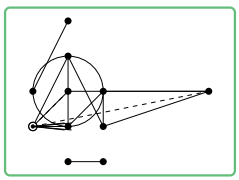
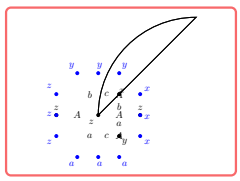
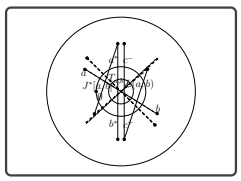
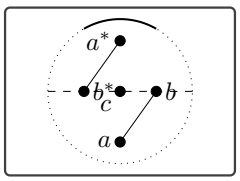
Reference	AutomaTikZ$_{v2}$	TikZero	TikZero+	GPT-4o
An illustration of the reduction from densest k-subgraph to u-rcp. On the left there is a simple undirected graph G with a single edge. The 2-reduced directed graph of G is on the right. Each vertex of G is replaced by $2 \cdot 2 = 4$ copies with a bidirectional edge connecting any two copies of the same vertex, and an outgoing edge from each copy to the single edge-vertex e.				
				
Domain of dependence: The closed trapezoidal region Ω_t used in the proof of proposition 1, shaded with darker yellow; the domain of dependence of point $(0, \ell)$ is Ω_t along with the top region shaded with lighter yellow.				
					
A sketch of iterating $f(x) = x - 1/x$. Points bigger than 1 get sent to point in $[0, 1]$ which then moves to a point ≤ -1 which moves to $[-1, 0]$ which is sent to a point ≥ 1 and so on.				
					
A multiline diagram D with $n = 6$ columns and $s = 5$ rows, with content $\lambda = (5, 4, 3, 1, 0, 0)$ and bottom row $\rho^{(1)}(D) = (4, 0, 1, 5, 3, 0) \in S_\lambda$. It has weight $w(D) = w_x(D)\, w_t(D) = x_1^3 x_2^2 x_3^4 x_4^2 x_5^2 x_6^2 t^2$.				
					

Table 8. Alternating rows display randomly selected tuples from the caption and image similarity human evaluation task (cf. Sec. 5.3). The frames of highest and lowest rated instances are highlighted in green and red, respectively.

```latex
\documentclass[border=10pt]{standalone}
\usepackage{pgfplots}
\pgfplotsset{width=7cm,compat=1.8}
\begin{document}
\begin{tikzpicture}
        \begin{axis}[grid=major,view={210}{30}]
        \addplot3+[domain=-5:5,surf]
                {exp(-x^2-y^2)*x};
        % connect classical points
        \addplot3+[mark=*,only marks,blue,samples
            at={-4.473,-1.463,1.463,4.473}]
                {exp(-x^2-y^2)*x};
        \end{axis}
\end{tikzpicture}
\end{document}
```

TikZero+

```latex
\documentclass[12pt]{article}
\usepackage{amsmath,amsthm,amssymb,amsfonts}
\usepackage{tikz}
\usepackage{pgfplots}
\usetikzlibrary{arrows.meta}

\begin{document}

\begin{tikzpicture}
\begin{axis}[view={60}{30}]
    \addplot3 [surf] {3*x^2 + y^2};
\end{axis}
\end{tikzpicture}

\end{document}
```

TikZero+

```latex
\documentclass[11pt,a4paper]{article}
\usepackage{amsmath}
\usepackage{amssymb}
\usepackage{color}
\usepackage{tikz}

\begin{document}

\begin{tikzpicture}[scale=0.5]
\draw [thick,->] (-2,0) -- (2,0);
\draw [thick,->] (0,-2) -- (0,2);
\draw [thick] (1.8,0) arc (0:180:1.8);
\node [right] at (2,0) {$\mathrm{Re}(\omega)$};
\node [above] at (0,2) {$\mathrm{Im}(\omega)$};
\node [right] at (1.8,1) {$\Gamma$};
\end{tikzpicture}

\end{document}
```

AutomaTikZ$_{v2}$

```latex
\documentclass{article}
\usepackage{amssymb}
\usepackage{amsmath}
\usepackage{pgfplots}
\pgfplotsset{compat=1.14}
\usepackage{tikz}
\usetikzlibrary{arrows}

\begin{document}

\begin{tikzpicture}[scale=3]
\draw[fill=green!15] (1,0) -- (2,0) -- (2,2) --
    (1,2) -- (1,0);
\draw[thick] (1,0) -- (1,2);
\draw[thick] (0,1) -- (2,1);
\node[below] at (1,0) {$0$};
\node[below] at (2,0) {$1$};
\node[left] at (1,2) {$1$};
\node[left] at (0,1) {$0$};
\node[above] at (1.5,1.5) {$\gamma$};
\draw[->,thick] (0,0) -- (1,0);
\draw[->,thick] (0,0) -- (0,1);
\draw[->,thick] (0,0) -- (0.5,0.5);
\draw[->,thick] (0,0) -- (1.5,0.5);
\draw[->,thick] (0,0) -- (0.5,1.5);
\draw[->,thick] (0,0) -- (1.5,1.5);
\draw[->,thick] (0,0) -- (2,0);
\draw[->,thick] (0,0) -- (0,2);
\draw[->,thick] (0,0) -- (1,1);
\end{tikzpicture}

\end{document}
```

AutomaTikZ$_{v2}$

Figure 6. Ti*k*Z programs generated by TikZero+ (top) and AutomaTikZ$_{v2}$ (LLM; bottom) corresponding to the figures shown in the first row of Fig. 1 in the same order.