

Collaborative use of KeTCindy with other mathematical tools

Masataka Kaneko

masataka.kaneko@phar.toho-u.ac.jp

Toho University, Japan

Satoshi Yamashita

sattch.yamashita@gmail.com

Kisarazu National College of Technology, Japan

Hideyo Makishita

hideyo@shibaura-it.ac.jp

Shibaura Institute of Technology, Japan

Koji Nishiura

nishiura@fukushima-nct.ac.jp

Fukushima National College of Technology, Japan

Setsuo Takato

takato@phar.toho-u.ac.jp

Toho University, Japan

Abstract

Interactive manipulation of mathematical objects on a PC screen and paper and pencil-based activities to process mathematical reasoning are both important aspects of mathematics education. In order to establish an effective linkage between these two sorts of activities, we have developed a plug-in named KeTCindy for the excellent dynamic geometry software Cinderella. KeTCindy converts graphical objects drawn with Cinderella into TeX readable code that generates the corresponding high-quality mathematical artwork in a PDF output.

To improve the graphics capabilities of the KeTCindy system, using it collaboratively with the symbolic computation capabilities of various computer algebra systems like Maxima or well-structured simulation capabilities of statistical software like R is desirable. Therefore, we have added the ability to invoke those computing software programs and importing the data calculated or simulated by them into Cinderella. Combining the imported data with the interactive

graphics capability of Cinderella should result in the ability to present an extremely wide range of mathematical objects.

In this paper, we will show some basic functions of the KeTCindy system together with some sample class materials generated with it. While showing them, the merits of the above mentioned collaborative use will be emphasized.

1 Introduction

To prepare teaching materials for mathematics, it is desirable to have various kinds of contents, such as mathematical expressions or formulas, scientific artwork, and tables of data, included in a unified manner. Moreover, to make simulations or experiments about some mathematical phenomena, refined computing tools should be involved. Since many convenient tools have been developed for each of these purposes, one possible way is to generate the necessary contents by using these tools and to create teaching materials by copying and pasting the contents from each. However, generating this way is often difficult since special copy functions, plain-text adjustments, or careful verification must often be used [1]. For instance, though \TeX is the most popular tool for generating high-quality mathematical expressions or formulas in printed college level teaching materials [2], it is not easy for most teachers to generate high-quality graphics or well-structured tables in \TeX documents. In fact, the inclusion of graphics generated with other graphical editors often causes some incompatibilities with the prescribed format of \TeX , and the tables in spreadsheets are difficult to be copied into \TeX documents. Thus many teachers have hoped for a way to convert the data for generating graphics or tables derived from other software into \TeX -readable code.

In order to facilitate the above mentioned conversion, the authors have developed KeTCindy [3] which is a plug-in to the popular dynamic geometry software Cinderella. It converts graphical objects drawn on the Cinderella screen into TeX readable code to generate the corresponding high-quality mathematical artwork in a PDF output. It also enables us to easily generate various tables in the preferred size and style utilizing the interactive graphics function of Cinderella. On the one hand, students and teachers can execute simulations or experiments concerning mathematical models to make conjectures or to establish conceptual images by using dynamic geometry software which allows them to display and handle graphical objects interactively on a PC screen [4][5]. On the other hand, students can make deductions or calculations which will help them to convert their understanding into their established knowledge through paper-and-pencil based activities with static graphics on printed materials [2][6]. Therefore KeTCindy can be expected to help students synchronize these two kinds of activities [7].

However, the capabilities of Cinderella and KeTCindy for algebraic computations or statistical simulations are not sufficient for detailed study on mathematical models. These capabilities are also needed in college mathematics education. To compensate for this inability, we have added to KeTCindy the ability to call up other computing tools like Maxima and R. Using this function, we can let these software compute the necessary data and import them into Cinderella. Further use of the imported data will lead to the production of teaching resources which are more refined mathematically. To develop this function, we fully utilized the scripting language of Cinderella named *CindyScript* [8]. Though the functions of algebraic computation and spreadsheets has been added to some dynamic geometry software tools like Geogebra [9], they are not as powerful as other tools

aiming at the same purpose.

In this paper, we will show some simple examples in which the above mentioned function of data exchange between Cinderella and other tools is utilized. These examples show that KeTCindy partially realizes the data exchange through workflow quite similar to copy-and-paste.

2 KeTCindy system

The procedure for generating mathematical artwork both on the Cinderella screen and on the \TeX final output through the KeTCindy system is summarized in Figure 1.

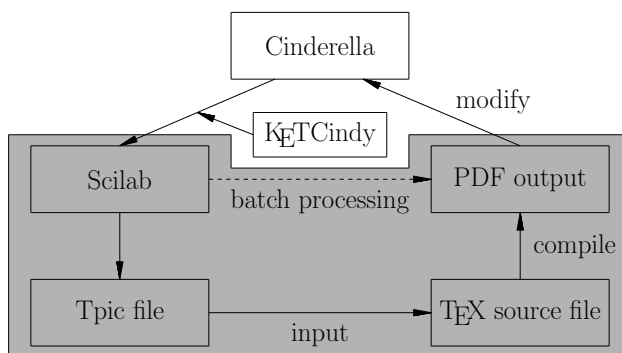


Figure 1: KeTCindy cycle

At first, we put geometric objects (such as points, segments, and circles) on the Cinderella screen and enter KeTCindy commands into CindyScript to generate the extra objects (such as graphs of functions, various accessories, and mathematical expressions) and control them on the screen. Then graphical images of mathematical objects are generated and manipulated interactively on the Cinderella screen, and KeTCindy converts the graphical data and associated commands into the commands of Scilab to generate the corresponding graphical data which are subsequently formatted into \TeX graphical codes. The compilation of a \TeX source file in which these \TeX graphical codes are input leads to the generation of high-quality graphical images in \TeX final outputs (in PDF form). To simplify the whole process, some batch processing displayed by a dashed arrow in Figure 1 has been added to the KeTCindy system. In fact, the processes included in the shaded region are executed simply by clicking a button on the Cinderella screen.

For example, Figure 2 shows a parabola $\frac{x^2}{a^2} - \frac{y^2}{b^2} = 1$ with parametric representation

$$x = a \cosh t \quad y = b \sinh t$$

and the region corresponding to $\left[t, t + \frac{1}{2} \right]$ displayed on the Cinderella screen. Let us consider the situation of giving students the task to prove that the area is independent of t .

In the bottom part of Figure 2, a “slider”, the blue segment, is generated. The x coordinate of the red point on this slider specifies t . By moving this point, students can observe how the position and shape of the shaded region change along with the change of t .

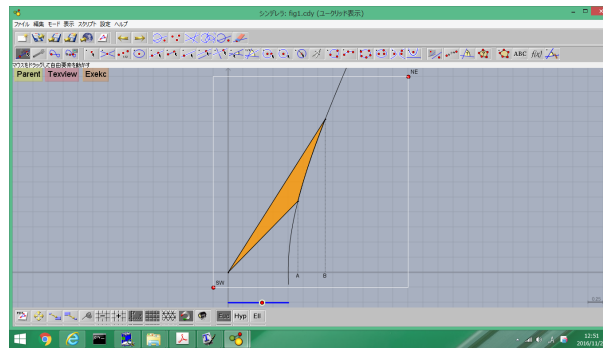


Figure 2: Cinderella screen

The KeTCindy commands needed to generate Figure 2 are shown in Figure 3.

```

1 Head="fig1";
2 Texparent="";
3 Keginit();
4 Paramplot("1", "(exp(t)+exp(-t))/2, (exp(t)-exp(-t))", "t=[-0.1, 1.3]", ["dr, 2"]);
5 O, xy=(0, 0); t=F, X;
6 P, xy=[(exp(t)+exp(-t))/2, (exp(t)-exp(-t))];
7 Q, xy=[(exp(t+1/2)+exp(-t-1/2))/2, (exp(t+1/2)-exp(-t-1/2))];
8 Listplot("1", [0, P]); Listplot("2", [0, Q]);
9 Parterv("1", P, Q, "gp1");
10 Enclosing("1", ["sg1", "part1", "invert(sg2)"]);
11 Listplot("3", [P, [P, x, 0]]); ["do"];
12 Listplot("4", [0, [0, x, 0]]); ["do"];
13 Shade(["en1"], [0.5, "color->hue(0.1)", "alpha->0.8"]);
14 Letter([P, x, 0], "s", "A"); Letter([0, x, 0], "s", "B");
15 Wndisp(gg);

generate Listplot sg2
generate partrv part1
generate Enclosing en1
generate Listplot sg1
generate Listplot sg4
output Shade of part1
    
```

Figure 3: KeTCindy commands in CindyScript

For students to prove it, paper-and-pencil based calculation of various regions is needed. Therefore, it is indispensable to have the same graphical image on printed matter. KeTCindy enables us to generate Figure 4 in the final $\text{T}_{\text{E}}\text{X}$ output simply by replacing the KeTCindy command

$$\text{Shade}(["en1"], [0.5, "color \rightarrow \text{hue}(0.1)", "alpha \rightarrow 0.8"]);$$

in Figure 3 with the command

$$\text{Hatchdata}("1", ["i"], [{"en1"}], [-45, 0.4]);$$

and by clicking two buttons “Texview” and “Exekek” on the Cinderella screen (see [S1]).

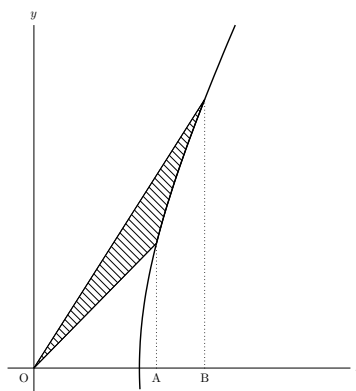


Figure 4: Final output in $\text{T}_{\text{E}}\text{X}$

Moreover, KeTCindy enables us to generate the corresponding animation on the final PDF output (see [S2]) which will be convenient when delivered to students who have not installed Cinderella on their own PCs. In fact, when downloading the PDF file, they can see the animation via Adobe reader. The KeTCindy commands needed in this case are quite simple as shown in Figure 5.

```

1 Fhead="fig1";
2 Texparent="";
3 Ketinit();
4 Ketinitmv();
5 Setunitlen("3cm");
6 Paramplot("1", "[[exp(t)+exp(-t)]/2, (exp(t)-exp(-t))]", "t=[-0.1, 1.6]", ["dr, 2"]);
7 O.xy=(0, 0);
8 Mf(t):=(
9 P.xy=[(exp(t)+exp(-t))/2, (exp(t)-exp(-t))];
10 O.xy=[(exp(t+1/2)+exp(-t-1/2))/2, (exp(t+1/2)-exp(-t-1/2))];
11 Listplot("1", [0, P]); Listplot("2", [0, 0]); Partorv("1", P, O, "gpl");
12 Enclosing("1", ["sg1", "part1", "invert(sg2)"]);
13 Shade(["en1", [0.5, "color->hue(0, 1)", "alpha->0.8"]);
14 Listplot("3", [P, [x, 0]]); Listplot("4", [O, [0, x, 0]]);
15);
16 Moviedata("Mf(s)", "s=[0, 1]", ["Div=40", "Cut=8"]);
17 Wndisp();

```

Figure 5: KeTCindy commands to generate animation

Taking full advantage of the programmability in CindyScript [8], we have implemented the function named `Moviedata` in Figure 5 to generate \TeX animation like this.

Also we can generate various tables in \TeX documents easily and flexibly by using KeTCindy. For instance, the table in Figure 6 can be generated on the \TeX final output by using the KeTCindy commands shown in Figure 7.

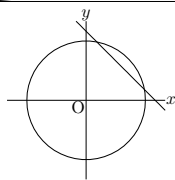
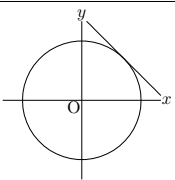
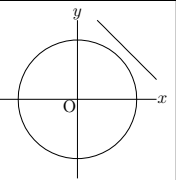
Answer			
Figure			
Discriminant	$D > 0$	$D = 0$	$D < 0$
solution	real number		imaginary number

Figure 6: Table generated on \TeX output

```

1 Fhead="table";
2 Texparent="";
3 Ketinit();
4 xLst=[23, 40, 40, 40]; yLst=[10, 40, 10, 10];
5 rmvL=["c2r0r1", "c3r0r1", "c3r0r1", "c2r3r4"];
6 Tabledata("", xLst, yLst, rmvL);
7 Putcell("c0r0", "c1r1", "c", "Answer");
8 Putcell("c0r1", "c1r2", "c", "Figure");
9 Putrow(2, "c", ["", "input[tablefig1]", "input[tablefig2]", "input[tablefig3]"]);
10 Putrow(3, "c", ["Discriminant", "$D>0$", "$D=0$", "$D<0$"]);
11 Putcell("c0r3", "c1r4", "c", "solution");
12 Putcell("c1r3", "c3r4", "c", "real number");
13 Putcell("c3r3", "c4r4", "c", "imaginary number");
14 Tlistplot("12u", ["c1r1", "c4r0"]); Tlistplot("12d", ["c1r0", "c4r1"]);
15 ChangeTablestyle(["r3lc4"], ["da"]);
16 Wndisp();

```

Figure 7: KeTCindy commands to generate the table

First we prepare three graphical files named “tablefigN” to depict all possible situations with discriminant. The function `Tabledata` has been implemented to generate the graphical images of tables. The sizes of each row and column are specified by using the following commands in Figure 7:

```
xLst=[23, 40, 40, 40]; yLst=[10, 40, 10, 10];
```

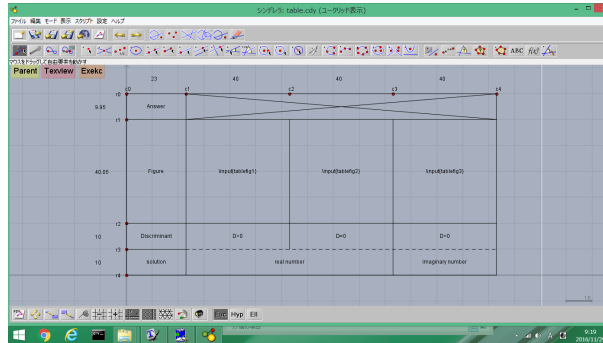


Figure 8: The graphical image of the table generated on the Cinderella screen

The KeTCindy package together with the related samples can be freely downloaded from the menu `Dropbox-Ketinstall` on our website.

3 Collaborative use with Maxima

When we use computer algebra systems (CASs), we often encounter the situation in which the result we obtain is represented differently as we expected. For example, if we use Maxima to calculate the indefinite integral $\int \frac{1}{1 + \sin x} dx$, we get the result in Procedure 1 below which is different from that obtained by a textbook approach in Procedure 2.

Procedure 1

$$\begin{aligned} \int \frac{1}{1 + \sin x} dx &= \int \frac{1 - \sin x}{(1 + \sin x)(1 - \sin x)} dx = \int \frac{1 - \sin x}{\cos^2 x} dx \\ &= \int \frac{1}{\cos^2 x} dx - \int \frac{\sin x}{\cos^2 x} dx = \tan x - \frac{1}{\cos x} \end{aligned}$$

Procedure 2

By substituting as $t = \tan \frac{x}{2}$,

$$\int \frac{1}{1 + \sin x} dx = \int \frac{1}{1 + \frac{2t}{1+t^2}} \frac{2}{1+t^2} dt = \int \frac{2}{(1+t)^2} dt = \frac{-2}{1+t} = \frac{-2}{1 + \tan \frac{x}{2}}$$

It would be effective if we could visualize both graphs (one entered directly and the other via appropriate call to Maxima) and then slide one of these graphs into the position of the other. This scenario can be easily realized using Cinderella and KeTCindy. As shown in Figure 9, KeTCindy is endowed with the function of letting Maxima execute the specified computation and importing the computed data into Cinderella which is subsequently used in the cycle of Figure 1 [10].

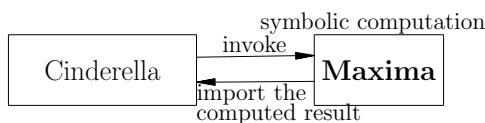


Figure 9: Collaborative use of KeTCindy with Maxima

Therefore, by using the KeTCindy commands in Figure 10, the following tasks can be executed.

1. Let Maxima compute the indefinite integral possibly along with Procedure 1 cited above and import the resulting representation into Cinderella via the function `Mxfun`
2. Draw the graphs of the function computed by Maxima and the function computed by hand possibly along with Procedure 2
3. Convert the imported representation into $\text{T}_{\text{E}}\text{X}$ readable form via the function `Mxtex`

```

1 Fhead="maxima";
2 Texparent="";
3 Koinit0();
4
5 f:= (1-sin(x))/cos(x)^2;
6 Mxfun("1", "integrate", [fn, "x"], [""]);
7 Plotdata("1", mx1, "x=[0, pi]", ["Num=200"]);
8 Mxtex("1", mx1); Expr([A, "c", tx1]);
9
10 Plotdata("2", "-2/(1+tan(x/2))+E.y+2", "x=[0, pi]", ["Num=200"]);
11 Expr([B, "c", "frac[-2][1+tan[x][2]]"]);
12
13 Windisp0();
14
    
```

Figure 10: The KeTCindy commands needed to collaborate with Maxima

As a result, it can be observed that the shape of the latter graph is the same as the former graph by using the slider on the Cinderella screen shown in Figure 11 (see [S3]).

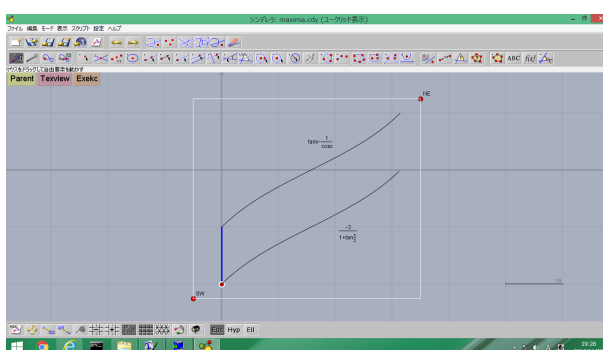


Figure 11: Graphs drawn on the Cinderella screen

Moreover, teachers can record this result by exporting the graphical image together with its mathematical representation into the $\text{T}_{\text{E}}\text{X}$ final output as shown in Figure 12. They can also copy the representation displayed on the console view at the bottom of the CindyScript screen and paste it onto any document in text format. Those documents can be easily delivered to students.

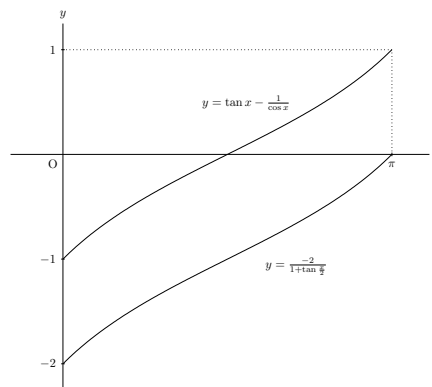


Figure 12: Graphs drawn on the \TeX final output

Thus, the whole computation process in Maxima is automatically activated and controlled by Cinderella. Furthermore, the computed data can be exchanged easily between Maxima (a computing tool), Cinderella (a simulating tool), and \TeX (a recording tool).

4 Collaborative use with Excel and R

Though we can easily make R import the data stored in Excel and execute the related simulations or statistical tests, it is more convenient if we can import the data directly to Cinderella as shown in Figure 13, since we often need to display them with the results of statistical analysis on printed matter.



Figure 13: Collaborative use of KeTCindy with Excel and R

To enable this workflow, we added the function `Gentable` to KeTCindy. In fact, its description is as follows:

```
Gentable(x) := (
  regional(dt, num1, num2, xLst, yLst, rmvL);
  dt=Tab2list(x);
  num1=length(dt_1); num2=length(dt);
  xLst=[]; yLst=[]; rmvL=[];
  repeat(num1, xLst=xLst++[A.x*10]; );
  repeat(num2, yLst=yLst++[B.y*10]; );
  Tabledatalight("1", xLst, yLst, rmvL, [2]);
  forall(1..num1, k,
    forall(1..num2, l,
      Putcell(k, l, "c", text(dt_1_k));
    );
  );
);
```


Then if we copy the table on the spreadsheet shown in Figure 14 and paste it into the part of `dtstr=""` in CindyScript as shown in Figure 15, the graphical image of the same table is generated both on the Cinderella screen (Figure 16) and on the final T_EX output (Figure 17) simply by pushing buttons. Its size can be arranged by moving the points A, B on the Cinderella screen.

tokyo	1	6	2
kyoto	4	2	6
osaka	5	3	7

Figure 14: Copying the data stored in Excel

```

1 Fhead="sample4";
2 Texparent="";
3 Ketinit();
4 Addax(0);
5
6 dtstr=""
7 tokyo 1 6 2
8 kyoto 4 2 6
9 osaka 5 3 7
10
11
12 Gentable(dtstr);
13
14 Windisp();
    
```

Figure 15: Pasting the data into CindyScript

tokyo	1	6	2
kyoto	4	2	6
osaka	5	3	7

Figure 16: Table generated on the Cinderella screen

tokyo	1	6	2
kyoto	4	2	6
osaka	5	3	7

Figure 17: Table generated on the final T_EX output

Once we have imported the data into Cinderella, we can also use it for an statistical analysis through the procedure shown in Figure 13. For instance, we can let R execute a χ^2 test based on the data in this cross table by using KeTCindy commands in Figure 18. Here the function `CalcbyR` makes R execute the statistical analysis specified in the part `cmdL=[]`; . As a result, we can obtain the resulting *p*-value displayed in the console view (see [S4]).

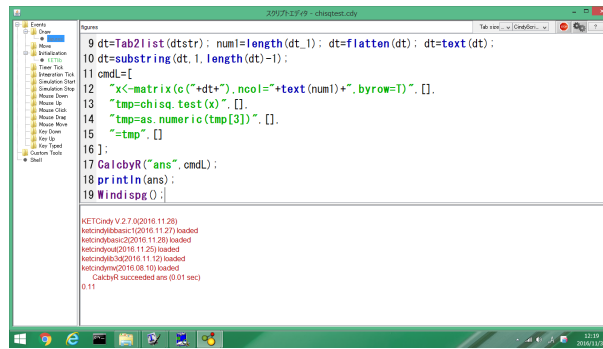


Figure 18: KeTCindy commands for collaborating with R

In a similar manner, we can make R execute statistical simulations and import the computed results for further use in graphics. For instance, executing the KeTCindy commands in Figure 19, we can generate random samples from a normal distribution and compute the realized values of statistic χ^2 . Also we can obtain the data for plotting the probability density function of χ^2 distribution and determine the critical region for the χ^2 test.

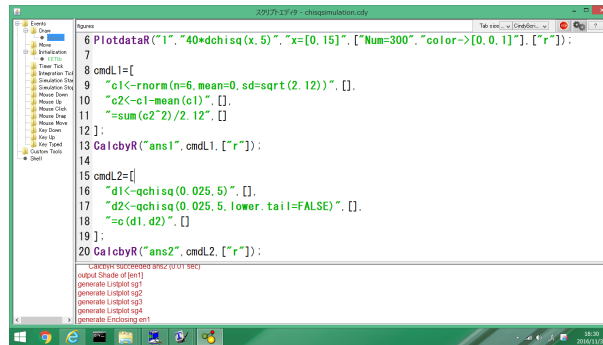


Figure 19: KeTCindy commands to let R execute a statistical simulation

Importing these data, we can generate the Cinderella screen as shown in Figure 20. Every time we push the button named “Simulation”, another simulation is executed and the resulting value of χ^2 is instantly displayed by a dashed line. Since the acceptance region for the χ^2 test is shaded in green, students can easily see whether or not the null hypothesis is accepted with the simulated result. Thus, by repeating the simulation many times, students can be expected to understand the precise meaning of the significant level. As in the previous cases, the graphical image of the simulated result can be generated readily on the T_EX output.

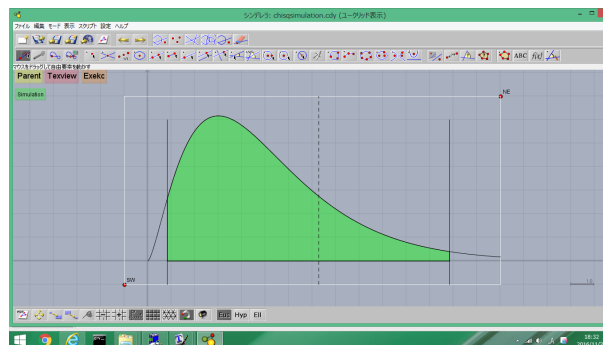


Figure 20: Cinderella screen used for simulation

5 Concluding remarks

As seen in the samples in this paper, the collaborative use of mathematical tools is effective and indispensable in some situations. In the framework of this paper, Cinderella plays the role of controller and translator between many tools via CindyScript. As a result, the exchange of data can be easily executed between those tools through a copy-and-paste like workflow.

6 Acknowledgements

This research is partially supported by Japan Society for the Promotion of Science (KAKENHI 15K01037).

References

- [1] M. Lokar: Lessons learned in course – computer tools in mathematics, Proceedings of CADGME2014
- [2] M. Kaneko, S. Takato: The effective use of LaTeX drawing in linear algebra – utilization of graphics drawn with KeTpic, The Electronic Journal of Mathematics and Technology 5-2, pp.129–148, 2012
- [3] M. Kaneko, S. Yamashita, K. Kitahara, Y. Maeda, Y. Nakamura, U. Kortenkamp, S. Takato: KeTCindy – Collaboration of Cinderella and KETpic, The International Journal for Technology in Mathematics Education 22-4, pp.179–185, 2015
- [4] A. Fest, M. Hiob, A. Hoffkamp: An interactive learning activity for the formation of the concept of function based on representation transfer, The Electronic Journal of Mathematics and Technology 5-2, pp.169–176, 2012
- [5] Z. Lavicza: Integrating technology into mathematics teaching: A review, ZDM: The International Journal of Mathematics Education 42-1, pp.105–119, 2010
- [6] S. Oviatt, A. Arthur, J. Cohen: Quiet interfaces that help students think, Proceedings of the 19th annual ACM Symposium on User Interface Software and Technology, pp.191–200, 2006
- [7] M. Kaneko: The actual use of KeTCindy in education, Lecture Notes in Computer Science 9725, pp.342–350, 2016
- [8] J. Richter-Gebert, U. Kortenkamp: The power of scripting: DGS meets programming, Acta Didactica Napocensia 3-2, pp.67–78, 2010
- [9] Z. Kovacs, B. Parisse: Giac and GeoGebra — Improved Grobner basis computations, Lecture Notes in Computer Science 8942, pp.126–138, 2015
- [10] S. Takato, A. McAndrew, M. Kaneko: Collaborative use of KeTCindy and free CASs for making materials, Proceedings of ACA2016, pp.69–73, 2016

Supplementary Electronic Material

- [S1] Basic procedure to generate $\text{T}_\text{E}\text{X}$ graphics via KeTCindy
- [S2] Procedure to generate $\text{T}_\text{E}\text{X}$ animation
- [S3] Collaboration between KeTCindy and Maxima
- [S4] Collabora<https://php.radford.edu/ejmt/>tion between KeTCindy and R