

פרויקט גמר- יישומי מחשב למדעים

מגישה- לינוי הורבט 312535248

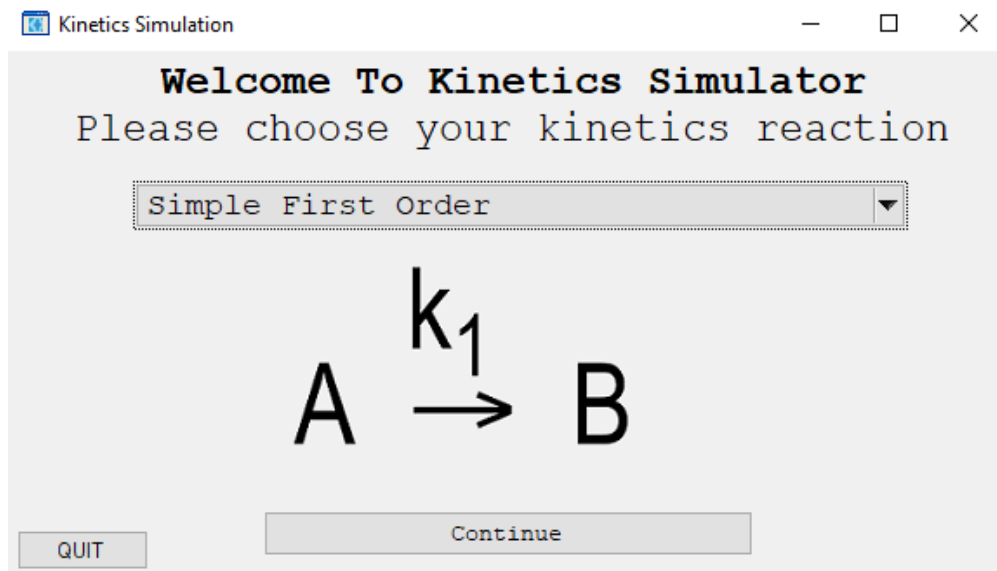
מטרת הפרויקט- סימולטור קינטיקה של תגובות שונות מסדר ראשון ושני.

משוואת הקצב בכימיה הינה כלי המאפשר לתאר את המהירות בה מתרחשת תגובה מסוימת. משוואת הקצב מקשרת בין קצב התגובה לבין ריכוזי המרכיבים והתוצרים, והיא מהווה כלי בסיסי וחשוב בהבנה הקינטית. הסימולטור הקינטי מאפשר חישוב ידידותי, מהיר, גרפי ונוח עבור תגובות קינטיות נפוצות וחשובות. הסימולטור בנוי משלושה מסכים: מסך ראשון בו המשתמש בוחר את סוג התגובה הרצויה, מסך שני בו המשתמש ממלא את הנתונים ומסך שלישי המציג את התוצאה.

הפנאל הראשון - מסך הכניסה

תכולת המסך:

- שני לחצני TEXT MESSAGE.
- שני לחצני RING.
- שני לחצני COMMAND BUTTON.

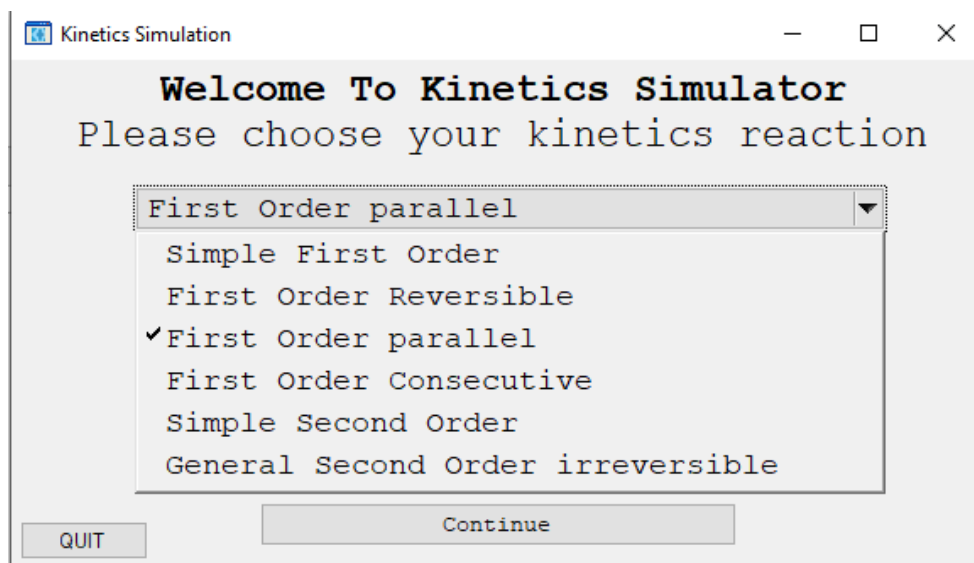


תמונה 1 מסך ראשי.

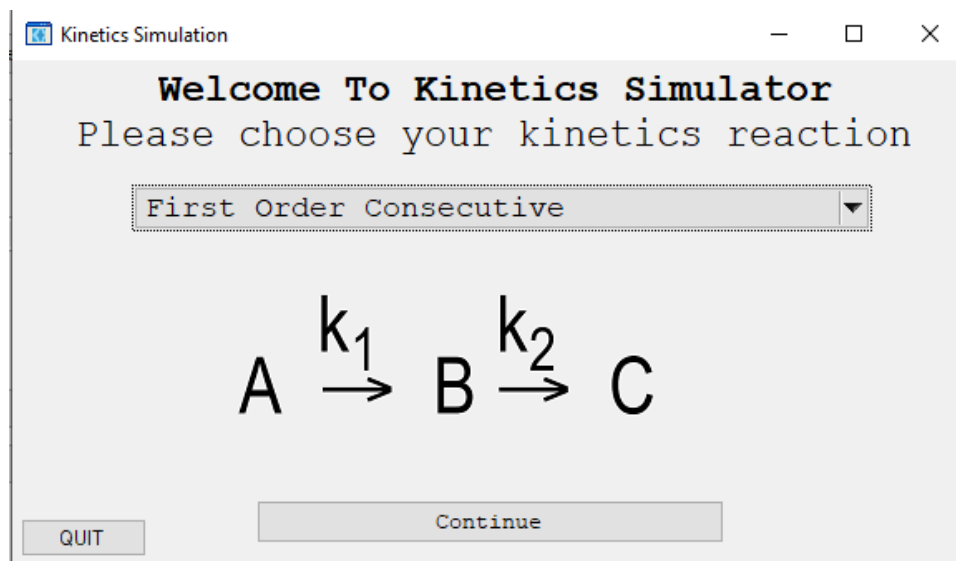
המסך הראשון הינו מסך התחלתי בו המשתמש בוחר את סוג התגובה הרצויה. במסך קיימים שני לחצני "הודעת טקסט" המכילים טקסט המברך ומנחה את המשתמש. כמו כן, המסך מכיל שני לחצני RING - לחצן המציג תמונה מתחלפת אוטומטית ולחצן בחירה אשר לחיצה עליו תגרום לפתיחת האפשרויות הקיימות לבחירת סוגי התגובות (ראה תמונה 2). התמונה היא

המשוואה הקינטית המתאימה לתיאור התגובה הנבחרת וכל בחירה של תגובה מסוימת גוררת שינוי אוטומטי שלה (ראה תמונה 3).

נוסף על כך המסך מכיל לחצן "המשך" המעביר את המשתמש למסך הבא וכן לחצן "יציאה" או סימן איקס בקצה המאפשרים לו לסגור את התוכנית.



תמונה 2 מסך ראשי - אפשרויות בחירת התגובה.

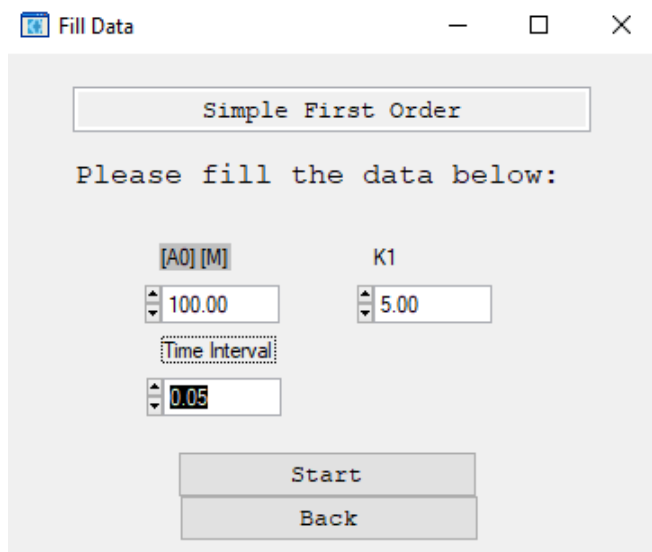


תמונה 3 מסך ראשי - דוגמה לשינוי משוואה כתלות בסוג התגובה.

הפנאל השני - מסך מילוי הנתונים

תכולת המסך:

- לחצן RING.
- שישה לחצני NUMERIC חלקם מוסתרים תחילה.
- שני לחצני COMMAND BUTTON.
- לחצן TIMER.



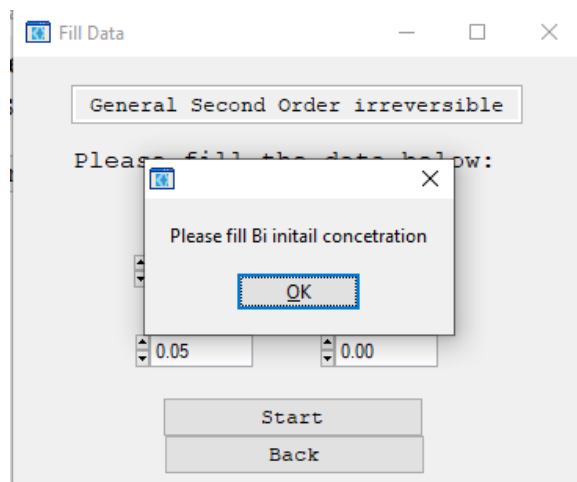
תמונה 4 מסך הזנת נתונים.

במסך זה המשתמש מזין את הנתונים ההתחלתיים לביצוע הסימולציה. לחצן ה-RING מראה את סוג התגובה ומשתנה באופן אוטומטי על פי בחירת המשתמש במסך הראשון. על המשתמש להזין את הנתונים המתאימים בלחצני ה-NUMERIC. הנתונים שיוזנו במסך זה משתנים בהתאם לסוג התגובה שנבחרה על ידי המשתמש במסך הקודם (ראה תמונות 4 ו-5). המשתמש חייב להזין את כל הנתונים בערכים תקינים על מנת להמשיך בסימולציה (ראה תמונה 9). אי הזנת נתון (ראה תמונות 6 ו-7) או לחילופין הזנת נתון בערך לא פסיקלי (ראה תמונה 8) לא יאפשרו למשתמש לעבור למסך הבא ויעלו הודעת שגיאה מתאימה.

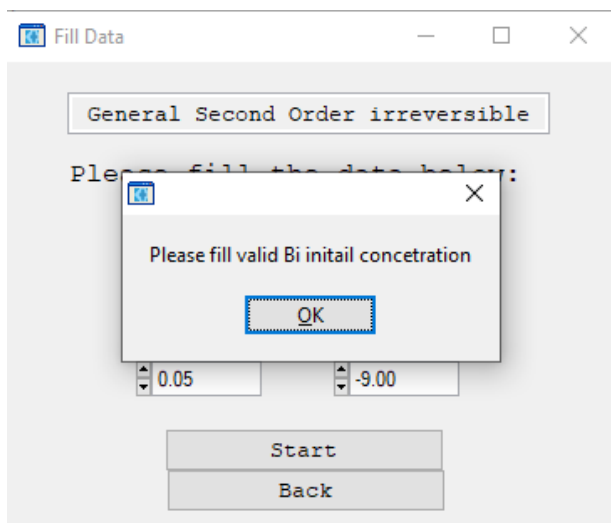
תמונה 5 מסך הזנת נתונים - דוגמה ללחצנים השונים הקיימים במסך זה כתלות בבחירת סוג התגובה במסך הראשון.

להלן מספר דוגמאות להודעות שגיאה אפשריות:

תמונה 6 מסך הזנת נתונים - הערת שגיאה עבור אי הזנת קבוע קצב התגובה. ההערה תתעדכן בהתאם לנתון החסר.

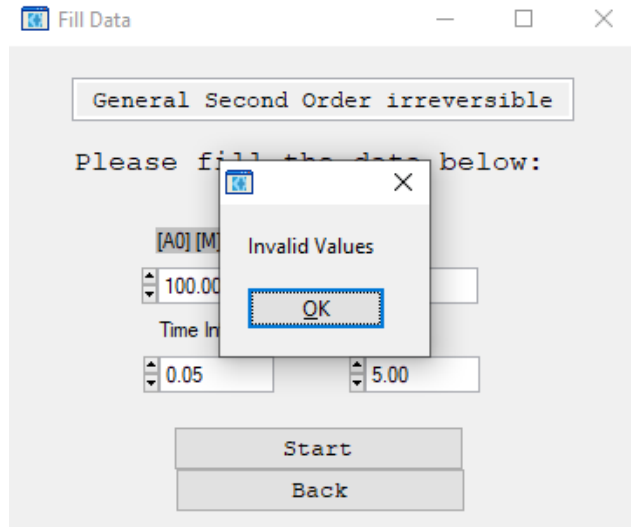


תמונה 7 מסך הזנת נתונים - דוגמה נוספת להערת שגיאה עבור ריכוז מגיב התחלתי חסר.



תמונה 8 מסך הזנת הנתונים - דוגמה להערת שגיאה עבור הזנת ערך לא פיסיקלי. ההערה תתעדכן בהתאם לנתון השגוי.

הודעת שגיאה נוספת הקיימת במערכת מתקבלת כאשר מוזנים ערכים לא פיסקליים. מאחר שאין כל משמעות פיסיקלית לריכוז שלילי, נתונים שיוזנו ויובילו בעקבות המשוואות בהמשך לקבלת ריכוז שלילי לא יתקבלו על ידי המערכת ויקפצו הודעת שגיאה מראש (ראה תמונה 9).



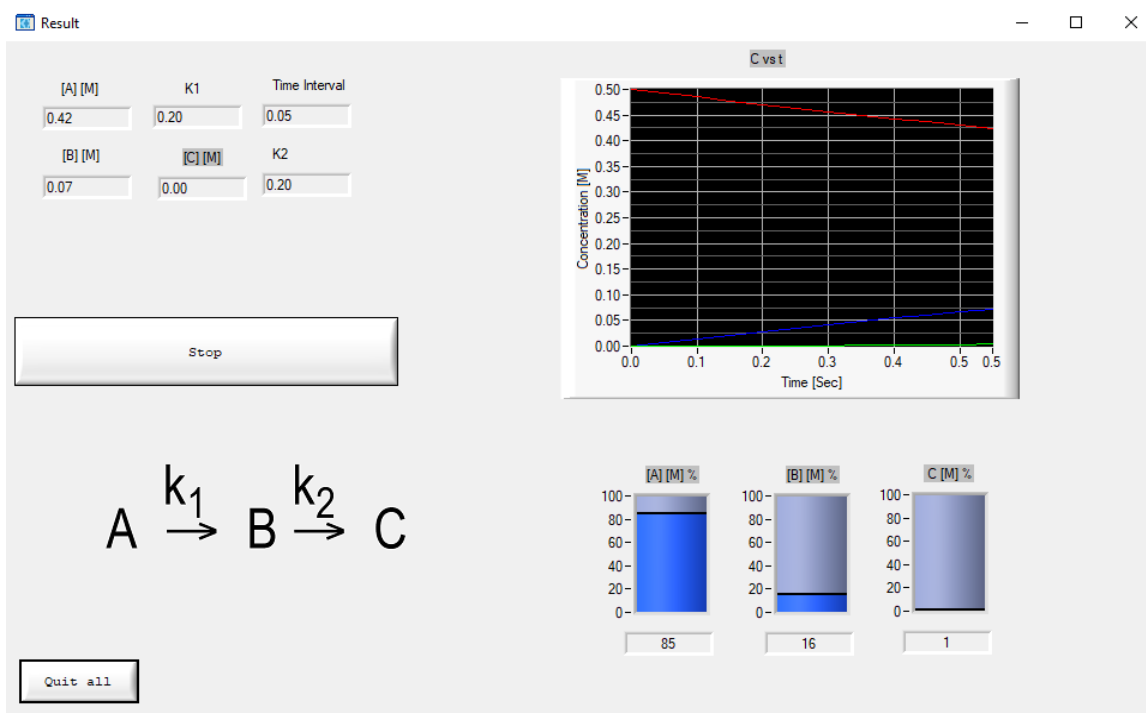
תמונה 9 מסך הזנת הנתונים - הערת שגיאה עבור הזנת נתונים לא פיסיקליים. הנתונים שהוזנו במסך זה יובילו בהמשך לקבלת ריכוז שלילי ולכן הוקפצה הודעה.

לאחר הזנת כל הנתונים המשתמש יכול לעבור למסך הבא בעזרת לחיצה על לחצן "התחל". לחצן זה יפעיל את הטיימר. כמו כן, לחצן "חזור" יאפשר למשתמש לחזור למסך הקודם.

פאנל שלישי - מסך התוצאה

המסך מכיל:

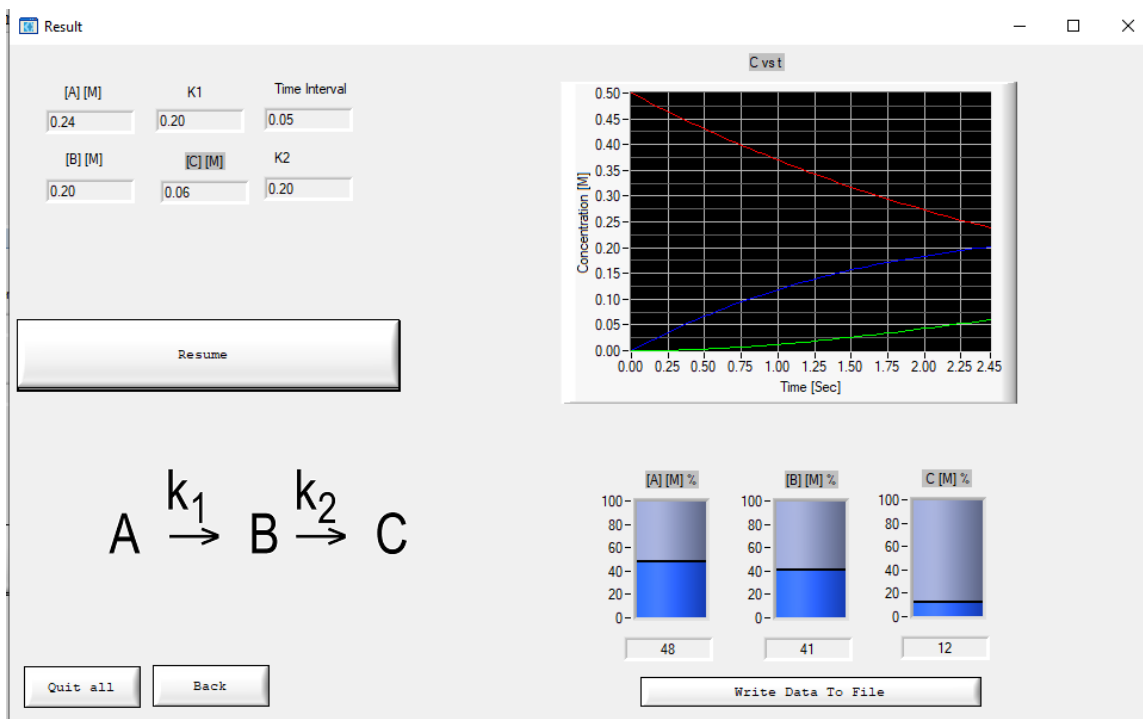
- לחצן RING.
- שמונה לחצני NUMERIC (חלקם מוסתרים תחילה).
- חמישה לחצני COMMAND BUTTON.
- לחצן GRAPH.
- לחצן BINARY SWITCH (מוסתר וגלוי כתלות בסוג התגובה).
- שלושה לחצני NUMERIC TANK (חלקם מוסתרים תחילה).



תמונה 10 מסך תוצאה.

בחלקו השמאלי העליון של מסך התוצאה מופיעים כל לחצני ה-NUMERIC שהוזנו במסך הקודם (מסך הזנת הנתונים) ועוד מספר לחצני NUMERIC משתנים בהתאם לתגובה שנבחרה. ערכי הלחצנים בחלק זה מתעדכנים באופן אוטומטי על פי התקדמות התגובה. מתחת ללחצנים אלו מופיע לחצן "עצור" אשר לחיצה עליו תעצור את הטיימר ותקפיא את מסך הנתונים. בנוסף כאשר לוחצים על לחצן זה יופיעו שלושה לחצנים נוספים: "המשך", "הקודם" ו-"כתוב נתונים לקובץ" (ראה תמונות 10 ו-11).

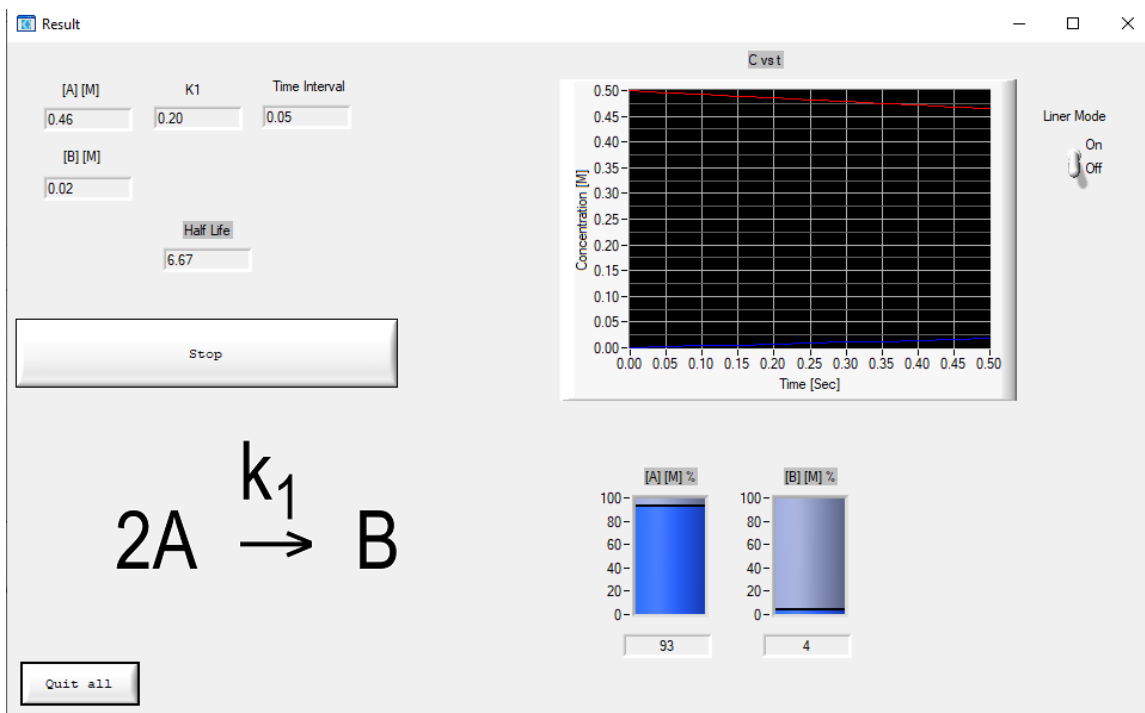
לחיצה על לחצן "המשך" תמשיך את התגובה מאותה נקודה שהוקפאה על ידי לחצן ה-"עצור" וכן תעלים חזרה את שלושת הלחצנים. לחצן "חזור" יחזיר את המשתמש למסך הקודם ויסגור מסך זה.



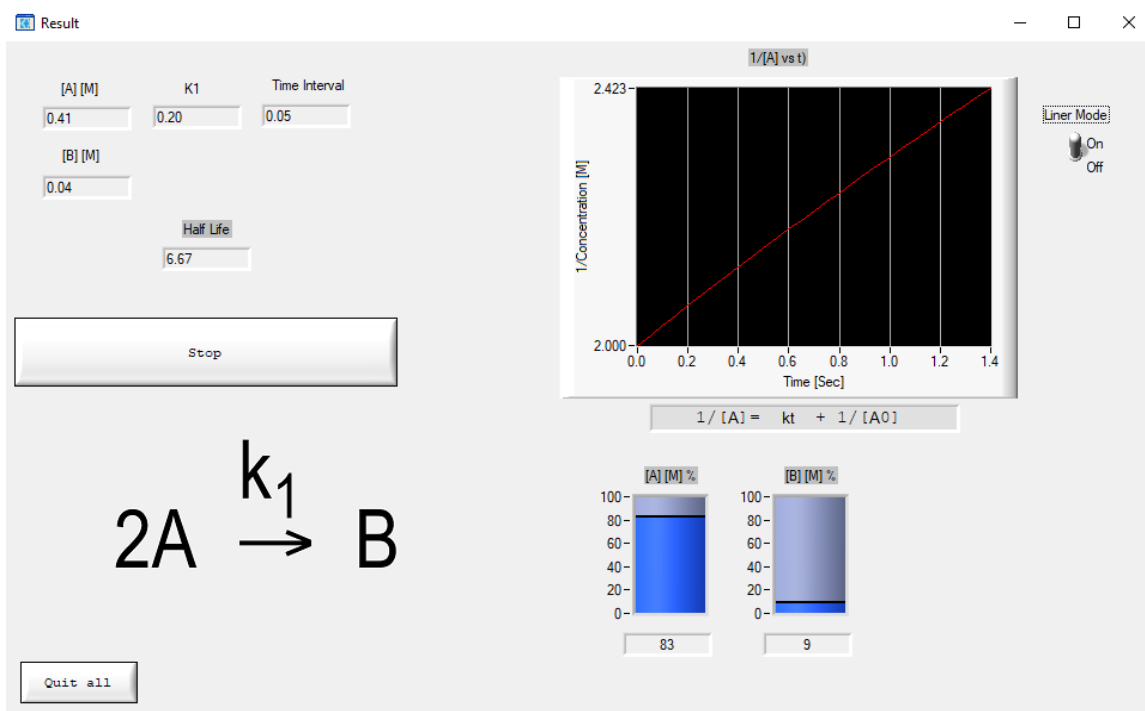
תמונה 11 מסך תוצאה - לחיצה על "עצור" והופעת האפשרויות: כתוב נתונים לקובץ, חזור והמשך.

מתחת ללחצנים אלו קיים לחצן תמונה מסוג **RING** בו מופיעה המשוואה המתארת את סוג התגובה שנבחרה במסך הראשון. המשוואה מתעדכנת באופן אוטומטי (ראה תמונות 11 ו-12). בתחתית העמוד מופיעה לחצן "יציאה" אשר סוגר את כל המסכים. בחלקו השמאלי העליון של המסך מופיע לחצן הגרף המתאר בזמן אמת את שינוי הריכוזים כפונקציה של הזמן. מתחת לגרף מופיעים לחצני ה-NUMERIC TANK, לחצני מיכל ויזואליים המתארים את שינוי הריכוזים באחוזים הן של התוצרים והן של המגיבים בזמן אמת לאורך התגובה.

עבור תגובות פשוטות מסדר ראשון ושני יופיע ערך זמן מחצית החיים. בנוסף יופיע לחצן בינארי המאפשר מעבר לתצוגת גרף ליניארית. לחיצה על לחצן זה תעלה גרף מתאים, תעדכן את כותרות הצירים ומתחתיו תופיע משוואת הגרף הרלוונטית. יודגש כי לחצן זה קיים עבור תגובות מסוג זה בלבד ואינו מופיע בשאר המצבים (ראה תמונות 12 ו-13).



תמונה 12 מסך תוצאה - דוגמה עבור מעבר בין גרף לינארי ולא לינארי בעזרת הכפתור הבינארי.



תמונה 13 מסך תוצאה - מעבר למצב לינארי בעזרת לחיצה על הכפתור הבינארי והופעת משוואת הגרף.

לחצן "כתוב נתונים לקובץ" אשר יופיע כאמור רק לאחר לחיצה על "עצור", יפתח קובץ חדש מסוג txt בשם "Kinetics Simulation Data.txt".

הקובץ יכיל כותרת ראשית, וכותרות משתנה המכילה את מספר הסימולציה. מתחת לכותרות המתאימות יופיעו הנתונים. הנתונים מסודרים בטבלה כאשר העמודה הראשונה היא הזמן, השנייה היא ריכוז המגיב ההתחלתי, ובהתאם לסוג התגובה, יופיעו ריכוזים נוספים עבור התוצרים (ראה תמונות 14 ו-15).

*Kinetics Simulation Data.txt - Notepad

File Edit Format View Help

Kinetics Simulation Data

Kinetics Simulation Data Run Number 1

[t]	[A]	[B]
0.000000	0.500000	0.000000
0.050000	0.492500	0.007500
0.100000	0.485113	0.014887
0.150000	0.477836	0.022164
0.200000	0.470668	0.029332
0.250000	0.463608	0.036392
0.300000	0.456654	0.043346
0.350000	0.449804	0.050196
0.400000	0.443057	0.056943
0.450000	0.436411	0.063589

Kinetics Simulation Data Run Number 2

[t]	[A]	[B]	[C]
0.000000	0.500000	0.000000	0.000000
0.050000	0.492500	0.007500	0.000000
0.100000	0.485113	0.014812	0.000075
0.150000	0.477836	0.021941	0.000223
0.200000	0.470668	0.028889	0.000443
0.250000	0.463608	0.035660	0.000731

Ln 1, Col 25 100% Windows (CRLF) UTF-8

תמונה 14 קובץ נתונים לדוגמה – בדוגמה זו המשתמש ביצע שני סימולציות של שני ראקציות שונות.

Kinetics Simulation Data.txt - Notepad

File Edit Format View Help

Kinetics Simulation Data

Kinetics Simulation Data Run Number 1

[t]	[A]	[B]
0.000000	0.500000	0.000000
0.050000	0.492500	0.007500
0.100000	0.485113	0.014887
0.150000	0.477836	0.022164
0.200000	0.470668	0.029332
0.250000	0.463608	0.036392
0.300000	0.456654	0.043346
0.350000	0.449804	0.050196
0.400000	0.443057	0.056943
0.450000	0.436411	0.063589

Kinetics Simulation Data Run Number 2

[t]	[A]	[B]	[C]
0.000000	0.500000	0.000000	0.000000
0.050000	0.492500	0.007500	0.000000
0.100000	0.485113	0.014812	0.000075
0.150000	0.477836	0.021941	0.000223
0.200000	0.470668	0.028889	0.000443
0.250000	0.463608	0.035660	0.000731

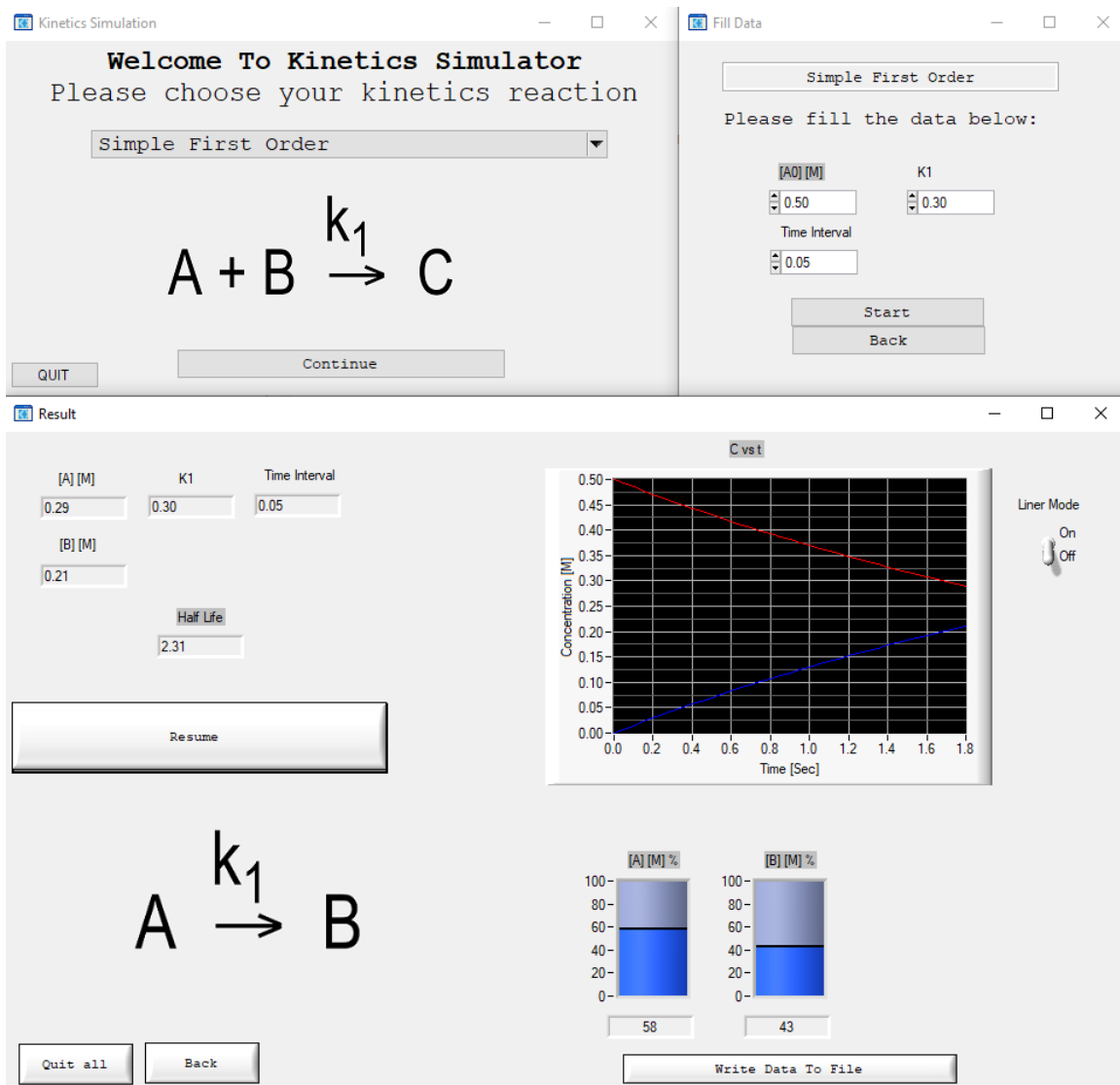
Kinetics Simulation Data Run Number 2

[t]	[A]	[B]	[C]
0.000000	0.500000	0.000000	0.000000
0.050000	0.492500	0.007500	0.000000
0.100000	0.485113	0.014812	0.000075
0.150000	0.477836	0.021941	0.000223
0.200000	0.470668	0.028889	0.000443
0.250000	0.463608	0.035660	0.000731
0.300000	0.456654	0.042258	0.001088
0.350000	0.449804	0.048685	0.001511
0.400000	0.443057	0.054945	0.001997
0.450000	0.436411	0.061042	0.002547

Ln 1, Col 1 100% Windows (CRLF) UTF-8

תמונה 15 קובץ נתונים לדוגמה - בדוגמה זו ראשית המשתמש לחץ על "עצור" ← "כתוב נתונים לקובץ" ← "המשך" " ← "כתוב נתונים לקובץ". לכן הקובץ מכיל פעמיים את אותה סימולציה, סימולציה מספר 2. בסימולציה 2 הראשונה מופיעים הנתונים עד העצירה הראשונה ובסימולציה 2 השנייה מופיעים כל הנתונים מתחילת הסימולציה עד העצירה השנייה.

דוגמת הרצה:



תמונה 16 דוגמת הצרת סימולציה

בדוגמה זו נבחרה במסך הראשון תגובה מסדר ראשון. במסך השני, ניתן לראות את הנתונים שהוזנו למערכת ואילו במסך השלישי את התוצאה שנוצרה.

להלן הסבר אודות החישובים עבור דוגמה זו:

כל הערכים שהמשתמש מזין נשמרים במשתנים גלובליים מתאימים במערכת מסוג double. הריכוזים של התוצר והמגיב משתנים בזמן על פי המשוואות הבאות:

$$[A]_{t2} = [A]_{t1} + dA_{t1}$$

$$[B]_{t2} = [B]_{t1} + dB_{t1}$$

כאשר :

$$dA_{t1} = -K_1[A]_{t1}dt$$

$$dB_{t1} = -dA_{t1}$$

זמן מחצית החיים במקרה זה מחושב על פי :

$$t_{0.5} = \frac{\ln(2)}{K_1}$$

ערכי המכלים הנומריים מחושבים על פי :

$$[A]_t = \frac{[A]_t}{[A]_0} * 100\%$$

$$[B]_t = \frac{[B]_t}{[A]_0} * 100\%$$

על מנת להציג את הנתונים בגרף ובקובץ חיצוני, קיימים שני משתנים גלובליים של מערך מסוג double. במשתנים אלה נשמרים כל ערכי הריכוזים שחושבו עד כה. למשל עבור דוגמה זו ישמרו המערכים הבאים :

$$A_arr=[0.5,...0.29]$$

$$B_arr=[0,...0.21]$$

$$time_arr=[0,0.05,...]$$

לחיצה על הלחצן הבינארי במקרה זה תגרום ליצירת מערך חדש - copy_A_arr. המערך יכיל את הערכים הבאים :

$$copy_A_arr=[\ln(A_arr[0]), \ln(A_arr[1]),...]$$

הגרף הלינארי יציג את המערך - copy_A_arr מול time_arr.

לחיצה על כפתור "כתוב נתונים לקובץ" תוביל לקבלת הקובץ הבא :

Kinetics Simulation Data.txt - Notepad

File Edit Format View Help

Kinetics Simulation Data

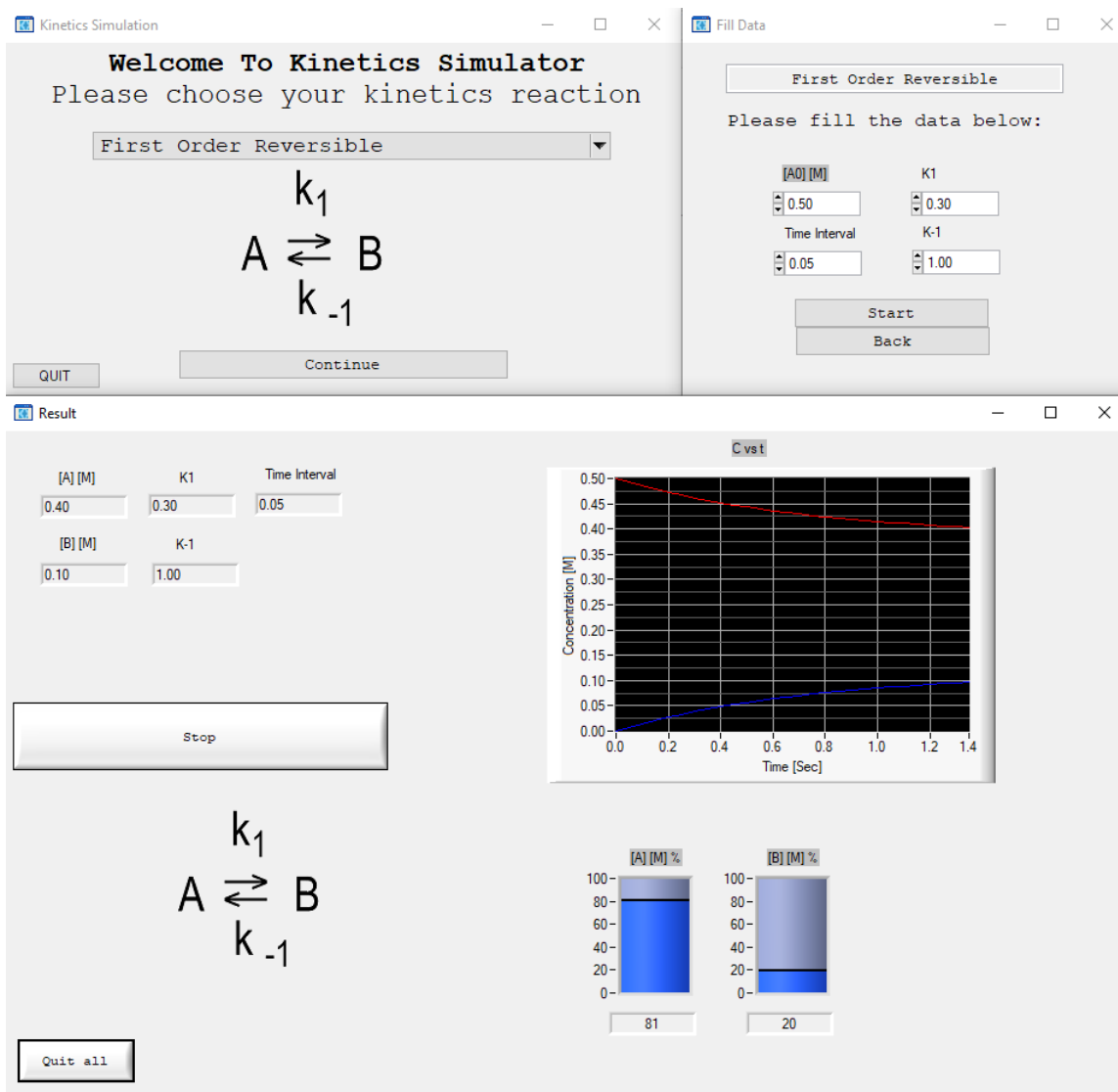
Kinetics Simulation Data Run Number 1

[t]	[A]	[B]
0.000000	0.500000	0.000000
0.050000	0.492500	0.007500
0.100000	0.485113	0.014887
0.150000	0.477836	0.022164
0.200000	0.470668	0.029332
0.250000	0.463608	0.036392
0.300000	0.456654	0.043346
0.350000	0.449804	0.050196
0.400000	0.443057	0.056943
0.450000	0.436411	0.063589
0.500000	0.429865	0.070135
0.550000	0.423417	0.076583
0.600000	0.417066	0.082934
0.650000	0.410810	0.089190
0.700000	0.404648	0.095352
0.750000	0.398578	0.101422
0.800000	0.392599	0.107401
0.850000	0.386710	0.113290
0.900000	0.380910	0.119090
0.950000	0.375196	0.124804
1.000000	0.369568	0.130432
1.050000	0.364025	0.135975
1.100000	0.358564	0.141436
1.150000	0.353186	0.146814
1.200000	0.347888	0.152112
1.250000	0.342670	0.157330
1.300000	0.337530	0.162470
1.350000	0.332467	0.167533
1.400000	0.327480	0.172520
1.450000	0.322568	0.177432
1.500000	0.317729	0.182271
1.550000	0.312963	0.187037
1.600000	0.308269	0.191731
1.650000	0.303645	0.196355
1.700000	0.299090	0.200910
1.750000	0.294604	0.205396
1.800000	0.290185	0.209815

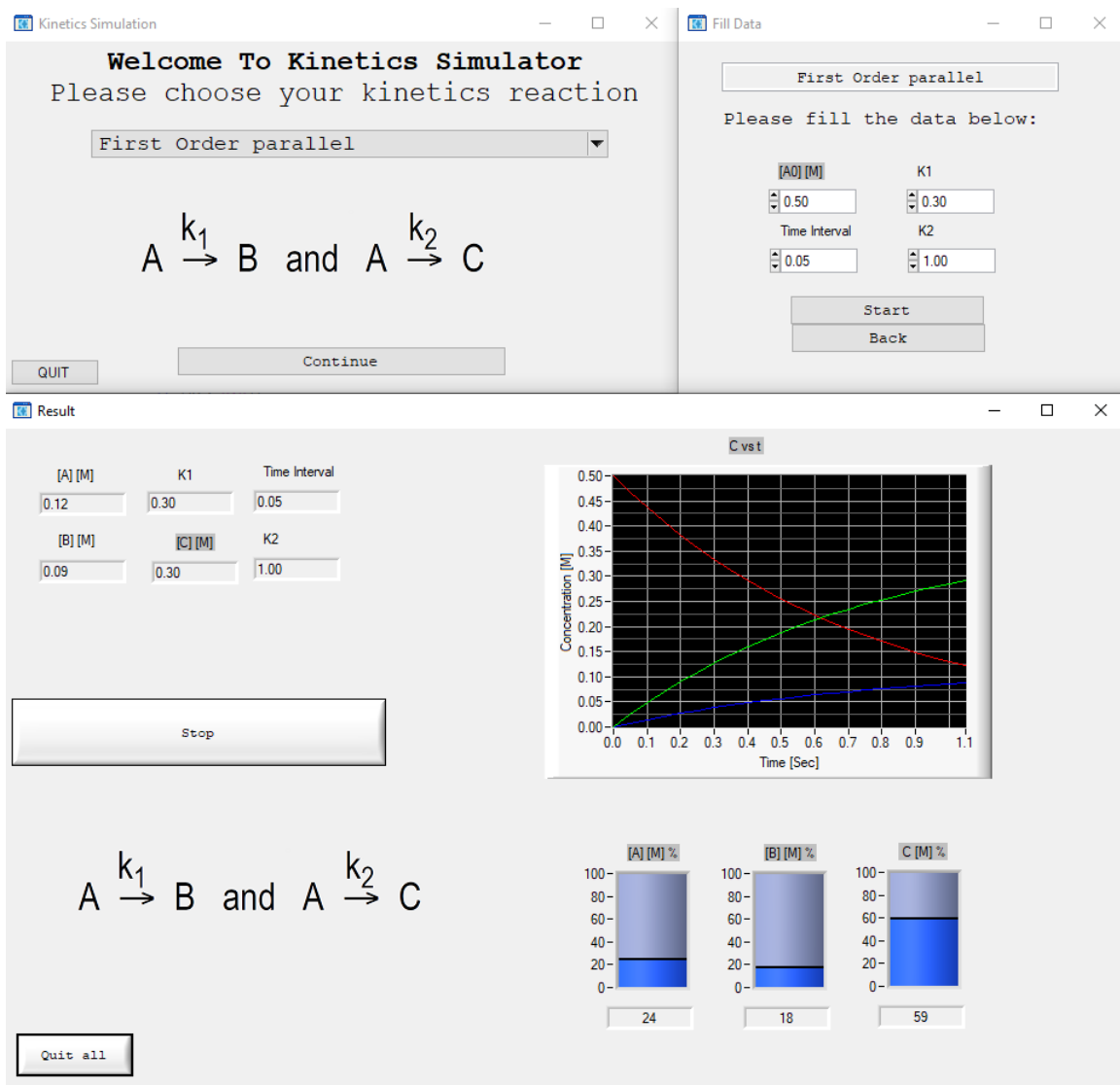
תמונה 17 דוגמת הרצה- קובץ מתקבל

בתום כל ריצה ובעת לחיצה על לחצן "חזור" קיימת פונקציה שמוחקת את הערכים שבמערכים מהזיכרון על מנת לא להעמיס.

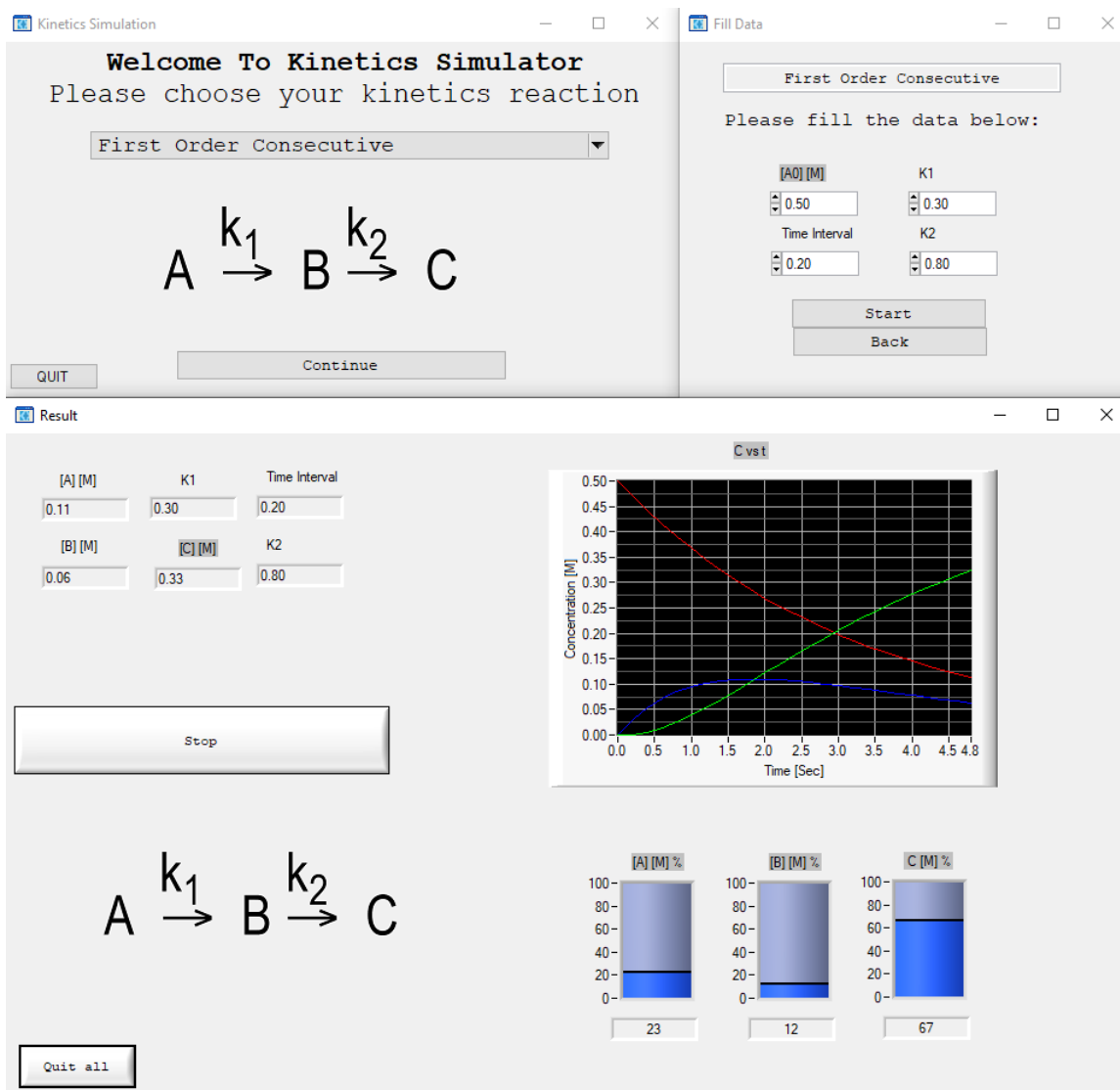
דוגמאות הרצה נוספת עבור שאר סוגי התגובות:



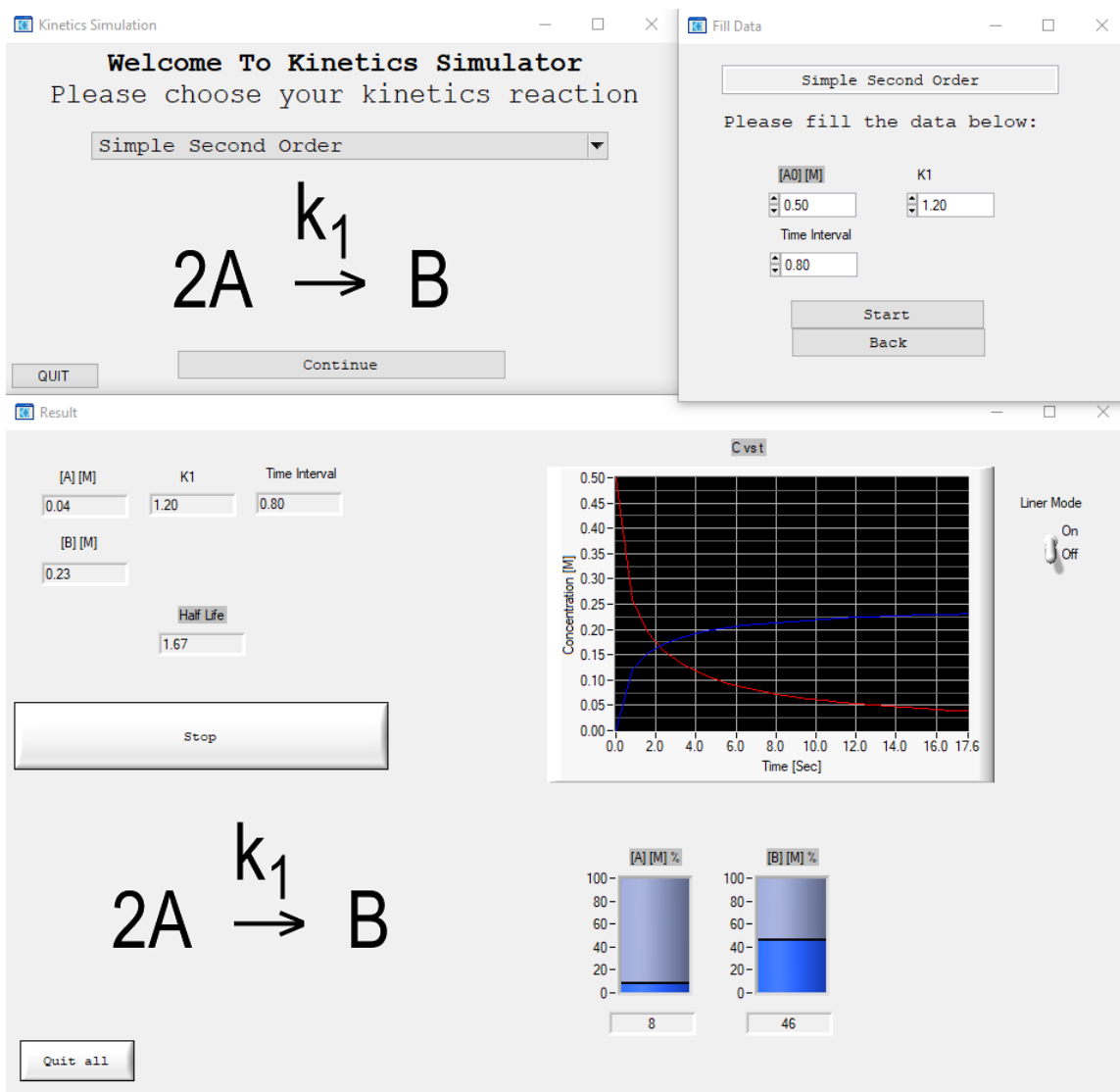
תמונה 18 דוגמת הרצה – עבור תגובה הפיכה מסדר ראשון.



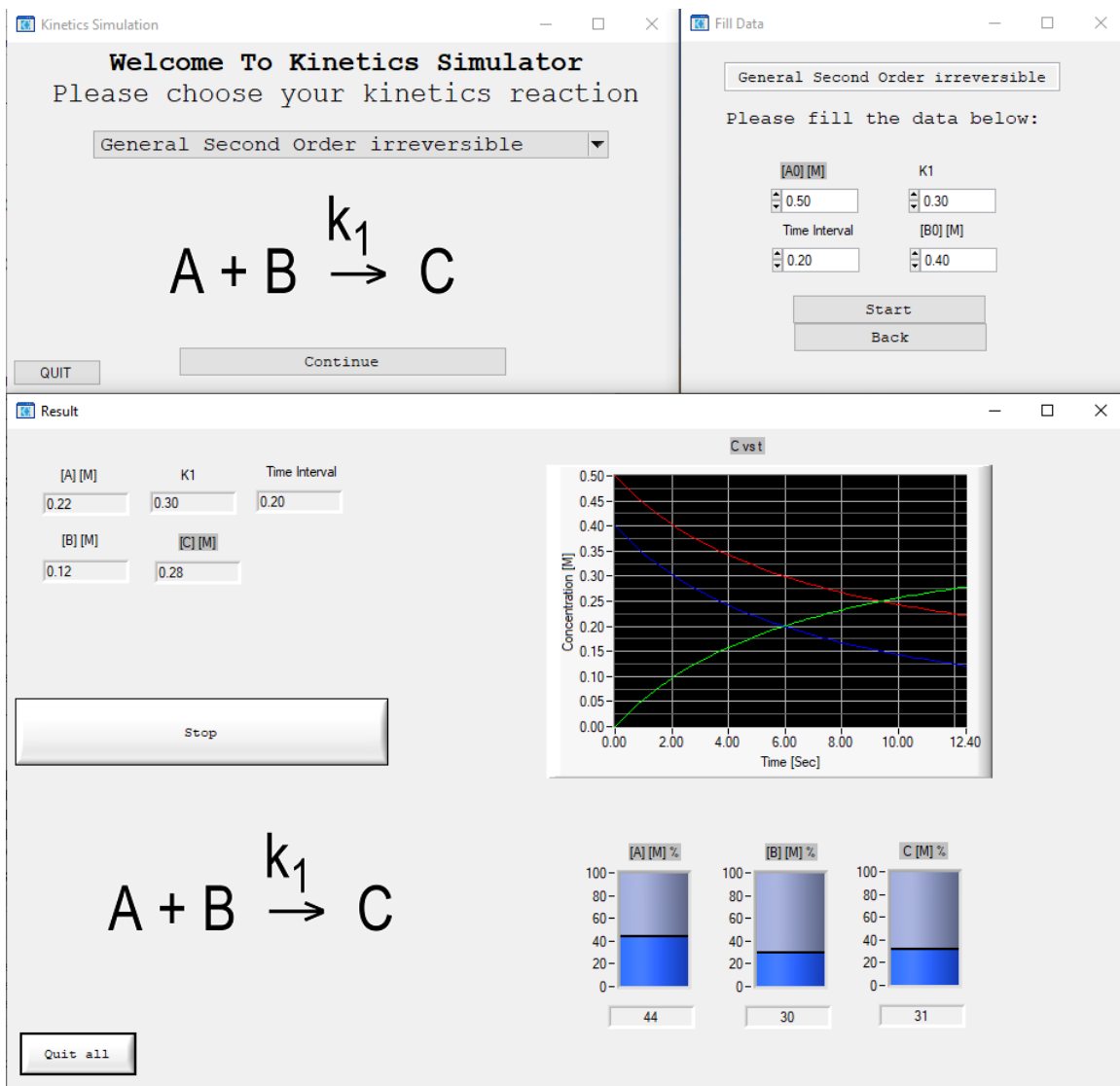
תמונה 19 דוגמת הרצה - עבור תגובה מקבילה מסדר ראשון.



תמונה 20 דוגמת הרצה - עבור תגובת שרשרת מסדר ראשון.



תמונה 21 דוגמת הרצה - עבור תגובה פשוטה מסדר שני.



תמונה 22 דוגמת הרצה - עבור תגובה פשוטה מסדר שני בלתי הפיכה.

```

#include <cvirte.h>
#include <userint.h>
#include "Kinetics Simulation.h"
#include <formatio.h>
#include <advanlys.h>
#include "progressbar.h"
#include "toolbox.h"
#include <cvirte.h>
#include <userint.h>
#define SIZE 10000

//-----Declare -----
double A,B,K1,K_1,K2,dt,C,half_T,Ai,Bi;
double
percent_A,percent_B,percent_C,A_arr[SIZE],B_arr[SIZE],time_arr[SIZE],C_arr[SIZE],copy_A_arr[SIZE];
double dA, dB, dC;
int count = 0;
FILE*fp;
char data[]="Kinetics Simulation Data.txt";
int countnumber=0;
static int panel1, panelHandle, panel2;
//-----Declare func-----
int CVICALLBACK graphfunc (int panel, int control, int event,
                           void *callbackData, int eventData1, int
eventData2);
//-----Clean Memo-----
-
void cleanMat(double mat[count], int count){
    for(int i=0;i<=count;i++){
        mat[i]=0;    }}
//-----MAIN FUNC-----
---
int main (int argc, char *argv[])
{
    if (InitCVIRTE (0, argv, 0) == 0)
        return -1;    /* out of memory */
    if ((panel1 = LoadPanel (0, "Kinetics Simulation.uir", PANEL_1)) < 0)
        return -1;
    DisplayPanel (panel1);
    RunUserInterface ();
    DiscardPanel (panel1);
    return 0;
}

```

```

//-----quit program-----
int CVICALLBACK quit (int panel, int control, int event,
                      void *callbackData, int eventData1, int
eventData2)
{
    switch (event)
    {
        case EVENT_COMMIT:

            A=B=K1=K_1=K2=dt=C=half_T=Ai=Bi=percent_A=percent_B=percent_C=0.0;
            QuitUserInterface (0);
            break;

    }
    return 0;
}
int CVICALLBACK finish2 (int panel, int event, void *callbackData,
                        int eventData1, int eventData2)
{
    switch (event)
    {
        case EVENT_GOT_FOCUS:

            break;
        case EVENT_LOST_FOCUS:

            break;
        case EVENT_CLOSE:

            QuitUserInterface (0);
            break;

    }
    return 0;
}
//-----panel1-----
int CVICALLBACK finish (int panel, int event, void *callbackData,
                       int eventData1, int eventData2)
{
    switch (event)
    {
        case EVENT_GOT_FOCUS:

            break;
        case EVENT_LOST_FOCUS:

            break;
        case EVENT_CLOSE:

```

```

        QuitUserInterface (0);
        break;
    }
    return 0;
}
int CVICALLBACK QuitCallback (int panel, int control, int event,
                             void *callbackData, int eventData1,
int eventData2)
{
    switch (event)
    {
        case EVENT_COMMIT:
            QuitUserInterface (0);
            break;
    }
    return 0;
}
int CVICALLBACK reactionfunc (int panel, int control, int event,
                             void *callbackData, int eventData1,
int eventData2)
{
    int ySwitch = 0;
    switch (event)
    {
        case EVENT_COMMIT:
            GetCtrlVal (panel1,PANEL_1_REAC_TYP, &ySwitch);
            switch (ySwitch)
            {
                case 1:
                    SetCtrlVal (panel1,PANEL_1_PICTUR_REC,1);
                    break;
                case 2:
                    SetCtrlVal (panel1,PANEL_1_PICTUR_REC,2);
                    break;
                case 3:
                    SetCtrlVal (panel1,PANEL_1_PICTUR_REC,3);
                    break;
                case 4:
                    SetCtrlVal (panel1,PANEL_1_PICTUR_REC,4);
                    break;
                case 5:
                    SetCtrlVal (panel1,PANEL_1_PICTUR_REC,5);
                    break;
                case 6:
                    SetCtrlVal (panel1,PANEL_1_PICTUR_REC,6);
                    break;
            }
    }
}

```

```

    }
        break;
    }
    return 0;
}
int CVICALLBACK continuefunc (int panel, int control, int event,
                             void *callbackData, int eventData1,
int eventData2)
{
    int ySwitch = 0;
    switch (event)
    {
        case EVENT_COMMIT:
            panel2=LoadPanel (0, "Kinetics Simulation.uir", PANEL_2);
            DisplayPanel(panel2);
            GetCtrlVal (panel1,PANEL_1_REAC_TYP, &ySwitch);
            switch (ySwitch)
            {
                case 1:
                    SetCtrlVal    (panel2, PANEL_2_RING,1);
                    break;
                case 2:
                    SetCtrlVal    (panel2, PANEL_2_RING,2);
                    SetCtrlAttribute (panel2, PANEL_2_RATE_k_1,
ATTR_VISIBLE,1);
                    break;
                case 3:
                    SetCtrlVal    (panel2, PANEL_2_RING,3);
                    SetCtrlAttribute (panel2, PANEL_2_RATE_k2,
ATTR_VISIBLE,1);
                    break;
                case 4:
                    SetCtrlVal    (panel2, PANEL_2_RING,4);
                    SetCtrlAttribute (panel2, PANEL_2_RATE_k2,
ATTR_VISIBLE,1);
                    break;
                case 5:
                    SetCtrlVal    (panel2, PANEL_2_RING,5);
                    break;
                case 6:
                    SetCtrlVal    (panel2, PANEL_2_RING,6);
                    SetCtrlAttribute (panel2, PANEL_2_conc_B,
ATTR_VISIBLE,1);
                    break;
            }
        break;
    }
}

```

```

    }
    return 0;
}
//-----panel2-----

int CVICALLBACK startfunc (int panel, int control, int event,
                           void *callbackData, int eventData1, int
eventData2)
{
    int ySwitch = 0;
    switch (event)
    {
        case EVENT_COMMIT:
            int a=1, b=1;
            GetCtrlVal (panel1, PANEL_1_REAC_TYP,&ySwitch);
            GetCtrlVal (panel2, PANEL_2_conc_A, &A);
            GetCtrlVal (panel2, PANEL_2_RATE_k1, &K1);
            Ai=A;
            GetCtrlVal (panel2, PANEL_2_TIME_INTERVAL, &dt);
            percnt_A=(A/Ai)*100.0;
            if (Ai<=0.0){
                b=0;
                if (Ai==0.0){
                    MessagePopup ("", "Please fill Ai initail concetration");}
                else {MessagePopup ("", "Please fill valid Ai initail
concentration");}
            }
            if(dt<=0.0){
                b=0;
                if (dt==0.0){
                    MessagePopup ("", "Please fill Time Interval");}
                else {MessagePopup ("", "Please fill valid Time
Interval");}
            }
            if(K1<=0.0){
                b=0;
                if (K1==0.0){
                    MessagePopup ("", "Please fill K1 Rate");}
                else {MessagePopup ("", "Please fill valid K1 Rate");}
            }
            if (a){
                switch (ySwitch)
                {
                    case 1:
                        if (K1*dt>=1){
                            b=0;

```



```

Values");}

PANEL_2_RATE_k_1, &K_1);

"Please fill K_1 Rate");}

("", "Please fill valid K_1 Rate");}

Values");}

PANEL_2_RATE_k2, &K2);

"Please fill K2 Rate");}

("", "Please fill valid K2 Rate");}

Values");}

PANEL_2_RATE_k2, &K2);

"Please fill K2 Rate");}

("", "Please fill valid K2 Rate");}

```

```

MessagePopup ("", "Invalid

break;

case 2:
GetCtrlVal (panel2,

if (K_1<=0.0){
    if (K_1==0.0){
        MessagePopup ("",

    else {MessagePopup

        b=0;
    }
    if ((K1-K_1)*dt>=1){
        b=0;
        MessagePopup ("", "Invalid

        break;

case 3:
GetCtrlVal (panel2,

if (K2<=0.0){
    if (K2==0.0){
        MessagePopup ("",

    else {MessagePopup

        b=0;
    }
    if ((K1-K2)*dt>=1){
        b=0;
        MessagePopup ("", "Invalid

        break;

case 4:
GetCtrlVal (panel2,

if (K2<=0.0){
    if (K2==0.0){
        MessagePopup ("",

    else {MessagePopup

        b=0;
    }
}

```

```

        if (K1*dt>=1){
            b=0;
            MessagePopup ("", "Invalid
Values");}

        break;

    case 5:
        if (K1*A*dt>=1){
            b=0;
            MessagePopup ("", "Invalid
Values");}

        break;

    case 6:
        GetCtrlVal (panel2,
PANEL_2_conc_B, &B);

        if (B<=0.0){
            b=0;
            if (B==0.0){MessagePopup
("", "Please fill Bi initail concetration");}

            else {MessagePopup ("",
"Please fill valid Bi initail concetration");}}

        if (K1*B*dt>=1){
            b=0;
            MessagePopup ("", "Invalid Values");}

        break;
    }}

    if (b) {
        switch (ySwitch)
        {
            case 2:
                GetCtrlVal (panel2, PANEL_2_RATE_k_1,
&K_1);

                break;

            case 3: case 4:
                GetCtrlVal (panel2, PANEL_2_RATE_k2, &K2);
                break;

            case 6:
                GetCtrlVal (panel2, PANEL_2_conc_B, &B);
                Bi=B;
                break;}

        panelHandle=LoadPanel(0, "Kinetics Simulation.uir",
PANEL);

        DisplayPanel(panelHandle);
        reactionfunc3 ( panelHandle, control, event,
                        callbackData, eventData1,
eventData2);

        countnumber++;

```

```

SetCtrlVal (panelHandle,
PANEL_NUMERICTANK_A,percnt_A);
SetCtrlAttribute (panel2,
PANEL_2_TIMER, ATTR_ENABLED, 1);}
    }
    return 0;
}
int CVICALLBACK timerfunc (int panel, int control, int event,
                           void *callbackData, int eventData1, int
eventData2)
{
    int ySwitch = 0;
    switch (event)
    {
        case EVENT_TIMER_TICK:
            GetCtrlVal (panel1, PANEL_1_REAC_TYP,&ySwitch);
            SetCtrlVal (panelHandle, PANEL_conc_A, A);
            percnt_A=(A/Ai)*100.0;
            SetCtrlVal (panelHandle, PANEL_NUMERICTANK_A,percnt_A);
            A_arr[count]=A;
            B_arr[count]=B;
            SetCtrlVal (panelHandle, PANEL_TIME_INTERVAL,dt);
            if (count==0)
                time_arr[count] =count;
            else
                time_arr[count]=time_arr[count-1]+dt;
            count++;
            SetCtrlAttribute (panelHandle, PANEL_NUMERICTANK_A,
ATTR_VISIBLE,1);
            SetCtrlVal (panelHandle, PANEL_RATE_k1,K1);
            graphfunc (panelHandle, control, event,
callbackData, eventData1, eventData2);
            switch (ySwitch)
            {
                case 1:
                    SetCtrlVal (panelHandle, PANEL_conc_B, B);
                    SetCtrlAttribute ( panelHandle, PANEL_conc_B,
ATTR_VISIBLE,1);
                    SetCtrlAttribute ( panelHandle,
PANEL_HALF_LIFE, ATTR_VISIBLE,1);
                    SetCtrlAttribute (panelHandle,
PANEL_NUMERICTANK_B, ATTR_VISIBLE,1);
                    dA=-K1*A*dt;
                    dB=-dA;
                    A=A+dA;
                    B=B+dB;

```

```

half_T=      log(2)/K1;
SetCtrlVal (panelHandle,
PANEL_HALF_LIFE,half_T);

percent_B=(B/Ai)*100.0;
SetCtrlVal (panelHandle,
PANEL_NUMERICTANK_B,percent_B);
break;
case 2:

SetCtrlVal (panelHandle, PANEL_conc_B, B);
SetCtrlVal (panelHandle,
PANEL_RATE_k_1,K_1);

SetCtrlVal (panelHandle, PANEL_RATE_k2,K2);
SetCtrlAttribute ( panelHandle, PANEL_conc_B,
ATTR_VISIBLE,1);

SetCtrlAttribute (panelHandle,
PANEL_NUMERICTANK_B, ATTR_VISIBLE,1);
dA=(-K1*A+K_1*B)*dt;
dB=-dA;
A=A+dA;
B=B+dB;

percent_B=(B/Ai)*100.0;
SetCtrlVal (panelHandle,
PANEL_NUMERICTANK_B,percent_B);
break;
case 3:

SetCtrlVal (panelHandle, PANEL_conc_B, B);
SetCtrlVal (panelHandle, PANEL_RATE_k2,K2);
SetCtrlAttribute ( panelHandle, PANEL_conc_B,
ATTR_VISIBLE,1);

SetCtrlAttribute ( panelHandle, PANEL_conc_C,
ATTR_VISIBLE,1);

SetCtrlAttribute (panelHandle,
PANEL_NUMERICTANK_B, ATTR_VISIBLE,1);
SetCtrlAttribute (panelHandle,
PANEL_NUMERICTANK_C, ATTR_VISIBLE,1);
dA=(-K1-K2)*A*dt;
dC=K2*A*dt;
dB=K1*A*dt;

A=A+dA;

B=B+dB;

C=C+dC;
C_arr[count]=C;
SetCtrlVal (panelHandle, PANEL_conc_C, C);
percent_B=(B/Ai)*100.0;
SetCtrlVal (panelHandle,
PANEL_NUMERICTANK_B,percent_B);

```

```

                                percent_C=(C/Ai)*100.0;
                                SetCtrlVal (panelHandle,
PANEL_NUMERICTANK_C,percent_C);
                                break;
                                case 4:
                                        SetCtrlVal (panelHandle, PANEL_conc_B, B);
                                        SetCtrlVal (panelHandle, PANEL_RATE_k2,K2);
                                        SetCtrlAttribute ( panelHandle, PANEL_conc_B,
ATTR_VISIBLE,1);
                                        SetCtrlAttribute ( panelHandle, PANEL_conc_C,
ATTR_VISIBLE,1);
                                        SetCtrlAttribute (panelHandle,
PANEL_NUMERICTANK_B, ATTR_VISIBLE,1);
                                        SetCtrlAttribute (panelHandle,
PANEL_NUMERICTANK_C, ATTR_VISIBLE,1);
                                        dA=-K1*A*dt;
                                        dB=(K1*A-K2*B)*dt;
                                        dC=K2*B*dt;
                                        A=A+dA;
                                B=B+dB;
                                        C=C+dC;
                                        SetCtrlVal (panelHandle, PANEL_conc_C, C);
                                        percent_B=(B/Ai)*100.0;
                                        SetCtrlVal (panelHandle,
PANEL_NUMERICTANK_B,percent_B);
                                        percent_C=(C/Ai)*100.0;
                                        SetCtrlVal (panelHandle,
PANEL_NUMERICTANK_C,percent_C);
                                        C_arr[count]=C;
                                break;
                                case 5:
                                        SetCtrlVal (panelHandle, PANEL_conc_B, B);
                                        SetCtrlAttribute ( panelHandle, PANEL_conc_B,
ATTR_VISIBLE,1);
                                        SetCtrlAttribute (panelHandle,
PANEL_NUMERICTANK_B, ATTR_VISIBLE,1);
                                        dA=-K1*A*A*dt;
                                        dB=-0.5*dA;
                                        A=A+dA;
                                B=B+dB;
                                        half_T=1/(K1*Ai);
                                        SetCtrlVal
(panelHandle,PANEL_HALF_LIFE,half_T);
                                        SetCtrlAttribute ( panelHandle,
PANEL_HALF_LIFE, ATTR_VISIBLE,1);
                                        percent_B=(B/Ai)*100.0;

```

```

        SetCtrlVal (panelHandle,
PANEL_NUMERICTANK_B,percnt_B);
        break;
        case 6:
            SetCtrlVal (panelHandle, PANEL_conc_B, B);
            SetCtrlAttribute ( panelHandle, PANEL_conc_B,
ATTR_VISIBLE,1);
            SetCtrlAttribute ( panelHandle, PANEL_conc_C,
ATTR_VISIBLE,1);
            SetCtrlAttribute (panelHandle,
PANEL_NUMERICTANK_B, ATTR_VISIBLE,1);
            SetCtrlAttribute (panelHandle,
PANEL_NUMERICTANK_C, ATTR_VISIBLE,1);
            dA=-K1*A*B*dt;
            dB=dA;
            dC=-dB;
            A=A+dA;
            B=B+dB;
            C=C+dC;
            C_arr[count]=C;
            SetCtrlVal (panelHandle, PANEL_conc_C, C);
            percnt_B=(B/Bi)*100.0;
            SetCtrlVal (panelHandle,
PANEL_NUMERICTANK_B,percnt_B);
            percnt_C=(C/(Ai+Bi))*100.0;
            SetCtrlVal (panelHandle,
PANEL_NUMERICTANK_C,percnt_C);
            break;
        }
        break;
    }
    return 0;
}
int CVICALLBACK back2_func (int panel, int control, int event,
                           void *callbackData, int eventData1,
int eventData2)
{
    switch (event)
    {
        case EVENT_COMMIT:
            DiscardPanel (panel2);
            DisplayPanel (panel1);
            break;
    }
    return 0;
}

```

```

//-----panel3-----
int CVICALLBACK reactionfunc3 (int panel, int control, int event,
                                void *callbackData, int
eventData1, int eventData2)
{
    int ySwitch = 0;
    switch (event)
    {
        case EVENT_COMMIT:
            GetCtrlVal (panel1,PANEL_1_REAC_TYP, &ySwitch);
            switch (ySwitch)
            {
                case 1:
                    SetCtrlVal
(panelHandle,PANEL_PICTUR_REC,1);
                    SetCtrlAttribute (panelHandle, PANEL_RATE_k2,
ATTR_VISIBLE,0);
                    SetCtrlAttribute (panelHandle, PANEL_conc_C,
ATTR_VISIBLE,0);
                    SetCtrlAttribute (panelHandle, PANEL_conc_B,
ATTR_VISIBLE,0);
                    SetCtrlAttribute (panelHandle,
PANEL_HALF_LIFE, ATTR_VISIBLE,0);
                    SetCtrlAttribute (panelHandle,
PANEL_RATE_k_1, ATTR_VISIBLE,0);
                    break;
                case 2:
                    SetCtrlVal
(panelHandle,PANEL_PICTUR_REC,2);
                    SetCtrlAttribute (panelHandle, PANEL_conc_B,
ATTR_VISIBLE,0);
                    SetCtrlAttribute (panelHandle,
PANEL_RATE_k_1, ATTR_VISIBLE,1);
                    SetCtrlAttribute (panelHandle, PANEL_RATE_k2,
ATTR_VISIBLE,0);
                    SetCtrlAttribute (panelHandle, PANEL_conc_C,
ATTR_VISIBLE,0);
                    SetCtrlAttribute (panelHandle,
PANEL_HALF_LIFE, ATTR_VISIBLE,0);
                    break;
                case 3:
                    SetCtrlVal
(panelHandle,PANEL_PICTUR_REC,3);
                    SetCtrlAttribute (panelHandle, PANEL_conc_B,
ATTR_VISIBLE,0);

```

```

        SetCtrlAttribute (panelHandle, PANEL_RATE_k2,
ATTR_VISIBLE,1);
        SetCtrlAttribute (panelHandle, PANEL_conc_C,
ATTR_VISIBLE,0);
        SetCtrlAttribute (panelHandle,
PANEL_RATE_k_1, ATTR_VISIBLE,0);
        SetCtrlAttribute (panelHandle,
PANEL_HALF_LIFE, ATTR_VISIBLE,0);
        break;
    case 4:
        SetCtrlVal
(panelHandle,PANEL_PICTUR_REC,4);
        SetCtrlAttribute (panelHandle, PANEL_conc_B,
ATTR_VISIBLE,0);
        SetCtrlAttribute (panelHandle, PANEL_RATE_k2,
ATTR_VISIBLE,1);
        SetCtrlAttribute (panelHandle, PANEL_conc_C,
ATTR_VISIBLE,0);
        SetCtrlAttribute (panelHandle,
PANEL_RATE_k_1, ATTR_VISIBLE,0);
        SetCtrlAttribute (panelHandle,
PANEL_HALF_LIFE, ATTR_VISIBLE,0);
        break;
    case 5:
        SetCtrlVal
(panelHandle,PANEL_PICTUR_REC,5);
        SetCtrlAttribute (panelHandle, PANEL_conc_B,
ATTR_VISIBLE,0);
        SetCtrlAttribute (panelHandle, PANEL_RATE_k2,
ATTR_VISIBLE,0);
        SetCtrlAttribute (panelHandle, PANEL_conc_C,
ATTR_VISIBLE,0);
        SetCtrlAttribute (panelHandle,
PANEL_HALF_LIFE, ATTR_VISIBLE,0);
        SetCtrlAttribute (panelHandle,
PANEL_RATE_k_1, ATTR_VISIBLE,0);
        break;
    case 6:
        SetCtrlVal
(panelHandle,PANEL_PICTUR_REC,6);
        SetCtrlAttribute (panelHandle, PANEL_conc_B,
ATTR_VISIBLE,1);
        SetCtrlAttribute (panelHandle, PANEL_conc_C,
ATTR_VISIBLE,1);
        SetCtrlAttribute (panelHandle, PANEL_RATE_k2,
ATTR_VISIBLE,0);

```



```

        SetCtrlAttribute (panelHandle,
PANEL_RATE_k_1, ATTR_VISIBLE,0);
        SetCtrlAttribute (panelHandle,
PANEL_HALF_LIFE, ATTR_VISIBLE,0);
        break;
    }
        break;
    }
    return 0;
}
int CVICALLBACK stopfunc (int panel, int control, int event,
void *callbackData, int eventData1, int
eventData2)
{
    int ySwitch = 0;
    switch (event)
    {
        case EVENT_COMMIT:
            SetCtrlAttribute (panel2, PANEL_2_TIMER, ATTR_ENABLED, 0);
            if (A_arr[0]!=0.0){
                SetCtrlAttribute (panelHandle, PANEL_FILE,
ATTR_VISIBLE,1);}
            GetCtrlVal (panel1,PANEL_1_REAC_TYP, &ySwitch);
            SetCtrlAttribute (panelHandle, PANEL_CONTINUE,
ATTR_VISIBLE,1);
            SetCtrlAttribute (panelHandle, PANEL_BACK,
ATTR_VISIBLE,1);
            switch (ySwitch){
                case 2:
                    for(int j=count-1;j>=0;j--){
                        copy_A_arr[j]=log(A_arr[j]-A);}
                    PlotXY (panelHandle, PANEL_GRAPH_2,
time_arr,copy_A_arr, count-1, VAL_DOUBLE, VAL_DOUBLE, VAL_THIN_LINE,
VAL_EMPTY_SQUARE,
VAL_SOLID, 1, VAL_RED);
                    break;
            }
            break;
    }
    return 0;
}
int CVICALLBACK resumefunc (int panel, int control, int event,
void *callbackData, int eventData1,
int eventData2)
{
    switch (event)

```

```

{
    case EVENT_COMMIT:
        SetCtrlAttribute (panelHandle, PANEL_FILE,
ATTR_VISIBLE,0);
        SetCtrlAttribute (panel2, PANEL_2_TIMER, ATTR_ENABLED,
1);
        SetCtrlAttribute (panelHandle, PANEL_CONTINUE,
ATTR_VISIBLE,0);
        SetCtrlAttribute (panelHandle, PANEL_BACK,
ATTR_VISIBLE,0);
        break;
    }
    return 0;
}

```

```

int CVICALLBACK writefile (int panel, int control, int event,
                           void *callbackData, int eventData1, int
eventData2)
{

```

```

    int ySwitch = 0;
    switch (event)
    {
        case EVENT_COMMIT:
            if (countnumber==1){
                fp=fopen(data,"w");
                fprintf (fp,"Kinetics Simulation Data");
                fclose(fp);}
            fp=fopen(data,"a");
            fprintf (fp,"\n\nKinetics Simulation Data Run Number
%d\n\n",countnumber);
            GetCtrlVal (panel1,PANEL_1_REAC_TYP, &ySwitch);
            switch (ySwitch){
                case 1: case 2: case 5:
                    fprintf (fp,"%-15s %-15s%-
15s\n","[t]","[A]","[B]");
                    for(int j=0;j<count;j++){
                        fprintf(fp,"%-15lf    %-15lf%-
15lf\n",time_arr[j],A_arr[j],B_arr[j]);}
                    break;
                case 3: case 4: case 6:
                    fprintf (fp,"%-15s %-15s%-15s%-
15s\n","[t]","[A]","[B]","[C]");
                    for(int j=0;j<count;j++){
                        fprintf(fp,"%-15lf    %-15lf%-15lf%-
15lf\n",time_arr[j],A_arr[j],B_arr[j],C_arr[j]);}
                    break;
            }
    }
}

```

```

        }
        fclose(fp);}
    return 0;
}

int CVICALLBACK graphfunc (int panel, int control, int event,
                           void *callbackData, int eventData1, int
eventData2)
{
    int ySwitch = 0;
    switch (event)
    {
        case EVENT_TIMER_TICK:
            GetCtrlVal (panel1, PANEL_1_REAC_TYP,&ySwitch);
            switch (ySwitch)
            {
                case 1:
                    SetCtrlAttribute (panelHandle,
PANEL_BINARYSWITCH, ATTR_VISIBLE,1);
                    PlotXY (panelHandle, PANEL_GRAPH_2,
time_arr,A_arr, count, VAL_DOUBLE, VAL_DOUBLE, VAL_THIN_LINE,
VAL_EMPTY_SQUARE,
VAL_SOLID, 1, VAL_RED );
                    SetCtrlAttribute (panelHandle, PANEL_GRAPH_2,
ATTR_YMAP_MODE, VAL_LOG);
                    PlotXY (panelHandle, PANEL_GRAPH, time_arr,
A_arr, count, VAL_DOUBLE, VAL_DOUBLE, VAL_THIN_LINE,
VAL_EMPTY_SQUARE,
VAL_SOLID, 1, VAL_RED);
                    PlotXY (panelHandle, PANEL_GRAPH, time_arr,
B_arr, count, VAL_DOUBLE, VAL_DOUBLE, VAL_THIN_LINE,
VAL_EMPTY_SQUARE,
VAL_SOLID, 1, VAL_BLUE);
                    break;

                case 2:
                    SetCtrlAttribute (panelHandle,
PANEL_BINARYSWITCH, ATTR_VISIBLE,0);
                    PlotXY (panelHandle, PANEL_GRAPH, time_arr,
A_arr, count, VAL_DOUBLE, VAL_DOUBLE, VAL_THIN_LINE,
VAL_EMPTY_SQUARE,
VAL_SOLID, 1, VAL_RED);
                    PlotXY (panelHandle, PANEL_GRAPH, time_arr,
B_arr, count, VAL_DOUBLE, VAL_DOUBLE, VAL_THIN_LINE,
VAL_EMPTY_SQUARE,
VAL_SOLID, 1, VAL_BLUE);
                    break;
            }
        }
    }
}

```

```

        case 3:
            SetCtrlAttribute (panelHandle,
PANEL_BINARYSWITCH, ATTR_VISIBLE,0);
            PlotXY (panelHandle, PANEL_GRAPH, time_arr,
A_arr, count, VAL_DOUBLE, VAL_DOUBLE, VAL_THIN_LINE,
VAL_EMPTY_SQUARE,
            VAL_SOLID, 1, VAL_RED);
            PlotXY (panelHandle, PANEL_GRAPH, time_arr,
B_arr, count, VAL_DOUBLE, VAL_DOUBLE, VAL_THIN_LINE,
VAL_EMPTY_SQUARE,
            VAL_SOLID, 1, VAL_BLUE);
            PlotXY (panelHandle, PANEL_GRAPH, time_arr,
C_arr, count, VAL_DOUBLE, VAL_DOUBLE, VAL_THIN_LINE,
VAL_EMPTY_SQUARE,
            VAL_SOLID, 1, VAL_GREEN);
            break;
        case 4:
            SetCtrlAttribute (panelHandle,
PANEL_BINARYSWITCH, ATTR_VISIBLE,0);
            PlotXY (panelHandle, PANEL_GRAPH, time_arr,
A_arr, count, VAL_DOUBLE, VAL_DOUBLE, VAL_THIN_LINE,
VAL_EMPTY_SQUARE,
            VAL_SOLID, 1, VAL_RED);
            PlotXY (panelHandle, PANEL_GRAPH, time_arr,
B_arr, count, VAL_DOUBLE, VAL_DOUBLE, VAL_THIN_LINE,
VAL_EMPTY_SQUARE,
            VAL_SOLID, 1, VAL_BLUE);
            PlotXY (panelHandle, PANEL_GRAPH, time_arr,
C_arr, count, VAL_DOUBLE, VAL_DOUBLE, VAL_THIN_LINE,
VAL_EMPTY_SQUARE,
            VAL_SOLID, 1, VAL_GREEN);
            break;
        case 5:
            SetCtrlAttribute (panelHandle,
PANEL_BINARYSWITCH, ATTR_VISIBLE,1);
            for(int j=0;j<count;j++){
                copy_A_arr[j]=1/A_arr[j];}
            PlotXY (panelHandle, PANEL_GRAPH_2,
time_arr,copy_A_arr, count, VAL_DOUBLE, VAL_DOUBLE, VAL_THIN_LINE,
VAL_EMPTY_SQUARE,
            VAL_SOLID, 1, VAL_RED);
            SetCtrlAttribute (panelHandle, PANEL_GRAPH_2,
ATTR_YMAP_MODE, VAL_LOG);
            PlotXY (panelHandle, PANEL_GRAPH, time_arr,
A_arr, count, VAL_DOUBLE, VAL_DOUBLE, VAL_THIN_LINE,
VAL_EMPTY_SQUARE,

```

```

        VAL_SOLID, 1, VAL_RED);
        PlotXY (panelHandle, PANEL_GRAPH, time_arr,
B_arr, count, VAL_DOUBLE, VAL_DOUBLE, VAL_THIN_LINE,
VAL_EMPTY_SQUARE,
        VAL_SOLID, 1, VAL_BLUE);
        break;

        case 6:
            PlotXY (panelHandle, PANEL_GRAPH, time_arr,
A_arr, count, VAL_DOUBLE, VAL_DOUBLE, VAL_THIN_LINE,
VAL_EMPTY_SQUARE,
            VAL_SOLID, 1, VAL_RED);
            PlotXY (panelHandle, PANEL_GRAPH, time_arr,
B_arr, count, VAL_DOUBLE, VAL_DOUBLE, VAL_THIN_LINE,
VAL_EMPTY_SQUARE,
            VAL_SOLID, 1, VAL_BLUE);
            PlotXY (panelHandle, PANEL_GRAPH, time_arr,
C_arr, count, VAL_DOUBLE, VAL_DOUBLE, VAL_THIN_LINE,
VAL_EMPTY_SQUARE,
            VAL_SOLID, 1, VAL_GREEN);
            break;}
    }

    return 0;
}

int CVICALLBACK back3_func (int panel, int control, int event,
                            void *callbackData, int eventData1,
int eventData2)
{
    switch (event)
    {
        case EVENT_COMMIT:
            stopfunc ( panel, control, event,
                        callbackData, eventData1, eventData2);
            if (A_arr[0]!=0.0){
                DiscardPanel (panelHandle);
                DisplayPanel (panel2);}
            if (count>0){
                cleanMat( A_arr, count);
                cleanMat( B_arr, count);
                cleanMat( C_arr, count);
                cleanMat( time_arr, count);
                cleanMat( copy_A_arr, count);
            }

            A=B=K1=K_1=K2=dt=C=half_T=Ai=Bi=percnt_A=percnt_B=percnt_C=dA=dB
            =dC=0.0;

```

```

        count = 0;
        break;
    }
    return 0;
}

int CVICALLBACK bin_func (int panel, int control, int event,
                          void *callbackData, int eventData1, int
eventData2)
{
    int switchVal = 0;
    int ySwitch = 0;
    switch (event)
    {
        case EVENT_COMMIT:
            GetCtrlVal (panel1, PANEL_1_REAC_TYP,&ySwitch);
            GetCtrlVal(panelHandle, PANEL_BINARYSWITCH,
&switchVal);

            if (switchVal == 1)
            {
                if (ySwitch==1){
                    SetCtrlVal(panelHandle, PANEL_EQ_LINER,1);
                    SetCtrlAttribute (panelHandle, PANEL_GRAPH_2,
ATTR_YNAME, "ln(Concentration [M])");
                }
                if (ySwitch==5){
                    SetCtrlVal(panelHandle, PANEL_EQ_LINER,2);
                    SetCtrlAttribute (panelHandle, PANEL_GRAPH_2,
ATTR_LABEL_TEXT, "1/[A] vs t");
                    SetCtrlAttribute (panelHandle, PANEL_GRAPH_2,
ATTR_YNAME, "1/Concentration [M]");}
                    SetCtrlAttribute (panelHandle, PANEL_GRAPH,
ATTR_VISIBLE,0);
                    SetCtrlAttribute (panelHandle, PANEL_GRAPH_2,
ATTR_VISIBLE,1);
                    SetCtrlAttribute (panelHandle, PANEL_EQ_LINER,
ATTR_VISIBLE,1);
                    SetCtrlAttribute (panelHandle, PANEL_GRAPH_2,
ATTR_XMAP_MODE, VAL_LINEAR);
                    SetCtrlAttribute (panelHandle, PANEL_GRAPH_2,
ATTR_YMAP_MODE, VAL_LINEAR);
                }
                else
                {
                    SetCtrlAttribute (panelHandle, PANEL_EQ_LINER,
ATTR_VISIBLE,0);

```

```

        SetCtrlAttribute (panelHandle, PANEL_GRAPH,
ATTR_VISIBLE,1);
        SetCtrlAttribute (panelHandle, PANEL_GRAPH_2,
ATTR_VISIBLE,0);
        SetCtrlAttribute (panelHandle, PANEL_GRAPH,
ATTR_XMAP_MODE, VAL_LINEAR);
        SetCtrlAttribute (panelHandle, PANEL_GRAPH,
ATTR_YMAP_MODE, VAL_LINEAR);
    }
    break;
}
return 0;
}

```