Московский авиационный институт
(Национальный исследовательский университет)
Факультет прикладной математики и информационных технологий
Кафедра вычислительной математики и программирования

Отчет по лабораторным работам

по курсу «Численные Методы»

Студент: Иларионов Д.А.
Группа: М8О-408Б-17
Преподаватель: Ревизников Д.Л.

Лабораторная работа 5

1. Тема ЛР:

   Численное решение уравнений параболического типа. Понятие о методе конечных разностей. Основные определения и конечно-разностные схемы. Решить уравнение параболического типа с помощью явной, неявной схемы и схемы Кранка-Никольсона.

2. Вариант : 7

   7.

   $$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + 0{,}5\exp(-0{,}5t)\cos x,$$

   $$u_x(0,t) = \exp(-0{,}5t),$$

   $$u_x(\pi,t) = -\exp(-0{,}5t),$$

   $$u(x,0) = \sin x,$$

   Аналитическое решение: $U(x,t) = \exp(-0{,}5t)\sin x$.

3. Алгоритм:

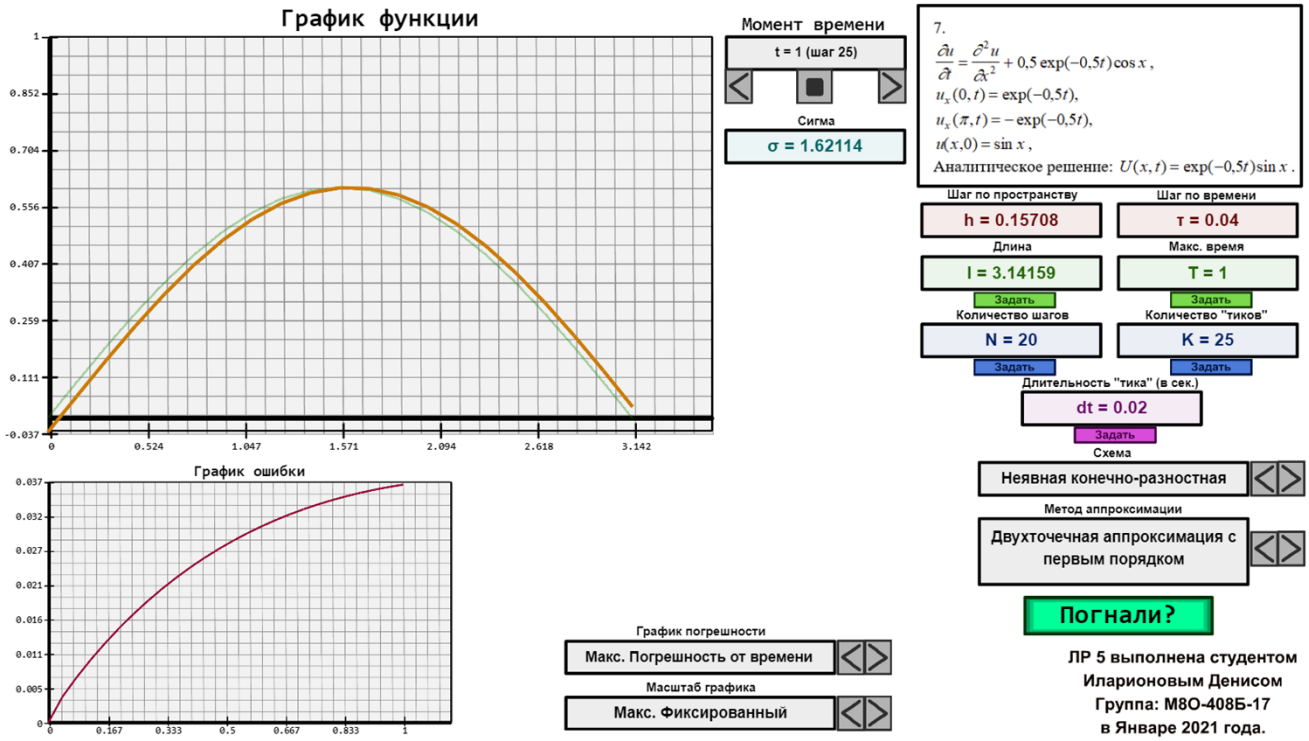
   Необходимо было решить уравнение мат. физики параболического тема с помощью конечно-разностных схем. Так как урматы я плохо понимал, ушло очень много времени на то, чтобы мне удалось что-то понять. В итоге, я смог реализовать алгоритмы. Трудной частью лабораторной также была аппроксимация граничных условий. С явной схемой все понятно, мы просто выводим формулу, потому что используются элементы прошлого временного слоя. С неявной немного труднее – у нас 3 неизвестных на каждой итерации. Но это можно решить с помощью метода прогонки, создав новую матрицу и подставив коэффициенты. С этим у меня была проблема и программа совсем плохо генерировала график. Кстати, самое главное – мы все уравнение разбиваем на блоки на двухмерной сетке. Такой принцип будет во всех лабораторных (а в последней и вовсе трехмерная сетка). Схема Кранка-Никольсона является комбинацией явной и неявной схемы и иногда дает лучше результаты, потому что явная схема часто опережает уравнение, а неявная – наоборот. Да, еще, явная схема является устойчивой, только тогда, когда число сигма (a*a*t/h^2) < 0.5. Где a – коэффициент перед производной. t – размер шага по времени, а h – размер шага по пространству. Неявная схема же всегда устойчива.

4. Среда разработки:

   Adobe Animate, Javascript + HTML5

5. Реализация

   Нарисовал и реализовал интерфейс для работы с уравнением. Можно задать шаги, скорость шага. Можно посмотреть погрешность, а также менять масштаб графика (динамический или статический).



6. Код (javascript)

```
var alpha = 1; //α
var beta = 0; //β
var gamma = 1; //γ
var delta = 0; //δ


//α * Ux(0, t) + β * U(0, t) = exp(-0.5*t)
//γ * Ux(l, t) + δ * U(l, t) = -exp(-0.5*t)
//Обозначим все как граничные условия 3 рода

function phi_0(t) {
    return Math.exp(-0.5 * t);
}

function phi_l(t) {
    return -Math.exp(-0.5 * t);
}

function psi(x) { return Math.sin(x); }
// Начальное условие



function U(x, t) {
    return Math.exp(-0.5 * t) * Math.sin(x);
} //Аналитическое решение


//Параметры сетки
var l = 3.1415926535; //интервал
var N = 10; //количество шагов
var T = 1; //макс. время
var K = 25; //количество тиков

var h = 0; var t = 0;

```

```
var a = 1; var b = 0; var c = 0;
//exp(-0.5t)cos(x) - доп. функция

var sigma = 1;
//σ = a^2 * τ / h^2

var dt = 0.02; //длительность тика
var cur_tick = 0;


var generation = 0; //номер генерации
var go = false; //увеличивать счетчик тиков?


this.addEventListener("tick", det_ht.bind(this));
function det_ht() {
    h = l / N;
    t = T / K;

    sigma = a * a * t / (h * h);
    //определяем шаги по времени и пространству

    this.h_text.text = "h = " + Math.round(h * 100000)/100000;
    this.tau_text.text = "τ = " + Math.round(t * 100000)/100000;

    this.l_text.text = "l = " + Math.round(l * 100000)/100000;
    this.T_text.text = "T = " + Math.round(T * 100000)/100000;

    this.N_text.text = "N = " + Math.round(N);
    this.K_text.text = "K = " + Math.round(K);

    this.dt_text.text = "dt = " + Math.round(dt*1000)/1000;

    this.sig_text.text = "σ = " + Math.round(sigma * 100000)/100000;


    if (go) {
        cur_tick += 1/60;
        if (cur_tick >= dt && s_cur < K) {
            var tGot = Math.floor(cur_tick / dt);
            cur_tick %= dt;
            s_cur += Math.min(K - s_cur, tGot);
        }
        else if (s_cur == K) {
            go = false;
        }
    }

}



this.setGreen1.addEventListener("click", set_l_prompt.bind(this));
function set_l_prompt() {
    generation += 1;
    var temp = prompt("Введите l:", '');
    temp = Number.parseFloat(temp);
    if (isNaN(temp)) {
        //если ввели какую-то хрень
        temp = l;
    }
    else if (temp > 1000000) {
                //ограничение на величину
                temp = 1000000;
            }
            else if (temp <= 0) {
                //всегда положительно
```

```
                temp = 0.001;
            }
            l = temp;
        }


        this.setGreen2.addEventListener("click", set_T_prompt.bind(this));
        function set_T_prompt() {
            generation += 1;
            var temp = prompt("Введите T:", '');
            temp = Number.parseFloat(temp);
            if (isNaN(temp)) {
                //если ввели какую-то хрень
                temp = T;
            }
            else if (temp > 1000000) {
                //ограничение на величину
                temp = 1000000;
            }
            else if (temp <= 0) {
                //всегда положительно
                temp = 0.001;
            }
            T = temp;
        }


        this.setBlue1.addEventListener("click", set_N_prompt.bind(this));
        function set_N_prompt() {
            generation += 1;
            var temp = prompt("Введите N:", '');
            temp = Number.parseInt(temp);
            if (isNaN(temp)) {
                //если ввели какую-то хрень
                temp = N;
            }
            else if (temp > 200) {
                //ограничение на величину
                temp = 200;
            }
            else if (temp <= 0) {
                //всегда положительно
                temp = 1;
            }
            N = temp;
        }


        this.setBlue2.addEventListener("click", set_K_prompt.bind(this));
        function set_K_prompt() {
            generation += 1;
            var temp = prompt("Введите K:", '');
            temp = Number.parseInt(temp);
            if (isNaN(temp)) {
                //если ввели какую-то хрень
                temp = K;
            }
            else if (temp > 10000) {
                //ограничение на величину
                temp = 10000;
            }
            else if (temp <= 0) {
                //всегда положительно
                temp = 1;
            }
            K = temp;
        }
```

```
173.
174.
175.        this.setPurp1.addEventListener("click", set_dt_prompt.bind(this));
176.        function set_dt_prompt() {
177.            cur_tick = 0;
178.            var temp = prompt("Введите dt (для ручного переключения можно ввести очень большим):",
    '');
179.            temp = Number.parseFloat(temp);
180.            if (isNaN(temp)) {
181.                //если ввели какую-то хрень
182.                temp = dt;
183.            }
184.            else if (temp > 1000000) {
185.                //ограничение на величину
186.                temp = 1000000;
187.            }
188.            else if (temp < 0.02) {
189.                //всегда положительно и больше 0.2 (почти плавная смена графика)
190.                temp = 0.02;
191.            }
192.            dt = temp;
193.        }
194.
195.
196.
197.        var scheme_type = 1;
198.        //Тип схемы
199.        //1,2 - Явная/Неявная К-Р схема
200.        //3 - схема Кранка-Николсона
201.
202.        var theta = 1;
203.
204.
205.        var meth_type = 1;
206.        //Метод аппроксимации
207.        //1 - Двухточечная аппроксимация с первым порядком
208.        //2 - Трехточечная аппроксимация со вторым порядком
209.        //3 - Двухточечная аппроксимация со вторым порядком
210.
211.
212.        var pogr_type = 1;
213.        //Тип погрешности
214.        //1 - Абсолютная
215.        //2 - Относительная
216.        //3 - От времени
217.
218.        var scale_type = 1;
219.        //Масштаб графика
220.        //1 - Фиксированный
221.        //2 - Динамический
222.
223.        this.SLeft1.addEventListener("click", scheme_left.bind(this));
224.        function scheme_left() {
225.            if (scheme_type == 1) {
226.                scheme_type = 3;
227.            }
228.            else {
229.                scheme_type -= 1;
230.            }
231.        }
232.
233.        this.SRight1.addEventListener("click", scheme_right.bind(this));
234.        function scheme_right() {
235.            if (scheme_type == 3) {
236.                scheme_type = 1;
237.            }
238.            else {
239.                scheme_type += 1;
```

```
            }
        }


        this.SLeft2.addEventListener("click", meth_left.bind(this));
        function meth_left() {
            if (meth_type == 1) {
                meth_type = 3;
            }
            else {
                meth_type -= 1;
            }
        }

        this.SRight2.addEventListener("click", meth_right.bind(this));
        function meth_right() {
            if (meth_type == 3) {
                meth_type = 1;
            }
            else {
                meth_type += 1;
            }
        }


        this.SLeft3.addEventListener("click", curStep_left.bind(this));
        function curStep_left() {
            if (s_cur > 0) {
                s_cur -= 1;
            }
        }

        this.SRight3.addEventListener("click", curStep_right.bind(this));
        function curStep_right() {
            if (s_cur < K) {
                s_cur += 1;
            }
        }


        this.stopBtn.addEventListener("click", curStep_stop.bind(this));
        function curStep_stop() {
            dt = 1000000;
        }

        this.SLeft4.addEventListener("click", pogr_left.bind(this));
        function pogr_left() {
            if (pogr_type == 1) {
                pogr_type = 3;
            }
            else {
                pogr_type -= 1;
            }
        }

        this.SRight4.addEventListener("click", pogr_right.bind(this));
        function pogr_right() {
            if (pogr_type == 3) {
                pogr_type = 1;
            }
            else {
                pogr_type += 1;
            }
        }

        this.SLeft5.addEventListener("click", scale_left.bind(this));
        function scale_left() {
            if (scale_type == 1) {
```

```
                scale_type = 2;
            }
            else {
                scale_type -= 1;
            }
        }

        this.SRight5.addEventListener("click", scale_right.bind(this));
        function scale_right() {
            if (scale_type == 2) {
                scale_type = 1;
            }
            else {
                scale_type += 1;
            }
        }

        this.addEventListener("tick", setTexts2.bind(this));
        function setTexts2() {
            if (scheme_type == 1) {
                this.scheme_text.text = "Явная конечно-разностная";
                theta = 1;
            }
            else if (scheme_type == 2) {
                this.scheme_text.text = "Неявная конечно-разностная";
                theta = 0;
            }
            else if (scheme_type == 3) {
                this.scheme_text.text = "Схема Кранка-Николсона";
                theta = 0.5;
            }

            if (meth_type == 1) {
                this.meth_text.text = "Двухточечная аппроксимация с первым порядком";
            }
            else if (meth_type == 2) {
                this.meth_text.text = "Трехточечная аппроксимация со вторым порядком";
            }
            else if (meth_type == 3) {
                this.meth_text.text = "Двухточечная аппроксимация со вторым порядком";
            }


            if (pogr_type == 1) {
                this.pogr_text.text = "Абсолютная погрешность";
            }
            else if (pogr_type == 2) {
                this.pogr_text.text = "Относительная погрешность";
            }
            else if (pogr_type == 3) {
                this.pogr_text.text = "Макс. Погрешность от времени";
            }


            if (scale_type == 1) {
                this.scale_text.text = "Макс. Фиксированный";
            }
            else if (scale_type == 2) {
                this.scale_text.text = "Динамический";
            }


            this.divDown0.text = "" + 0;
            this.divDown1.text = "" + Math.round(l*1/6*1000)/1000;
            this.divDown2.text = "" + Math.round(l*2/6*1000)/1000;
            this.divDown3.text = "" + Math.round(l*3/6*1000)/1000;
            this.divDown4.text = "" + Math.round(l*4/6*1000)/1000;
            this.divDown5.text = "" + Math.round(l*5/6*1000)/1000;
```

```
            this.divDown6.text = "" + Math.round(l*1000)/1000;


            if (pogr_type <= 2) {
                this.divDown0err.text = "" + 0;
                this.divDown1err.text = "" + Math.round(l*1/6*1000)/1000;
                this.divDown2err.text = "" + Math.round(l*2/6*1000)/1000;
                this.divDown3err.text = "" + Math.round(l*3/6*1000)/1000;
                this.divDown4err.text = "" + Math.round(l*4/6*1000)/1000;
                this.divDown5err.text = "" + Math.round(l*5/6*1000)/1000;
                this.divDown6err.text = "" + Math.round(l*1000)/1000;
            }
            else {
                this.divDown0err.text = "" + 0;
                this.divDown1err.text = "" + Math.round(T*1/6*1000)/1000;
                this.divDown2err.text = "" + Math.round(T*2/6*1000)/1000;
                this.divDown3err.text = "" + Math.round(T*3/6*1000)/1000;
                this.divDown4err.text = "" + Math.round(T*4/6*1000)/1000;
                this.divDown5err.text = "" + Math.round(T*5/6*1000)/1000;
                this.divDown6err.text = "" + Math.round(T*1000)/1000;
            }


            if (maxVals.length > 0 && scale_type == 2) {
                this.divUp0.text = "" + Math.round(minVals[s_cur]*1000)/1000;
                this.divUp1.text = "" + Math.round((minVals[s_cur] + (maxVals[s_cur] -
    minVals[s_cur])*1/7)*1000)/1000;
                this.divUp2.text = "" + Math.round((minVals[s_cur] + (maxVals[s_cur] -
    minVals[s_cur])*2/7)*1000)/1000;
                this.divUp3.text = "" + Math.round((minVals[s_cur] + (maxVals[s_cur] -
    minVals[s_cur])*3/7)*1000)/1000;
                this.divUp4.text = "" + Math.round((minVals[s_cur] + (maxVals[s_cur] -
    minVals[s_cur])*4/7)*1000)/1000;
                this.divUp5.text = "" + Math.round((minVals[s_cur] + (maxVals[s_cur] -
    minVals[s_cur])*5/7)*1000)/1000;
                this.divUp6.text = "" + Math.round((minVals[s_cur] + (maxVals[s_cur] -
    minVals[s_cur])*6/7)*1000)/1000;
                this.divUp7.text = "" + Math.round(maxVals[s_cur]*1000)/1000;
            }
            else {
                this.divUp0.text = "" + Math.round(minMinVal*1000)/1000;
                this.divUp1.text = "" + Math.round((minMinVal + (maxMaxVal -
    minMinVal)*1/7)*1000)/1000;
                this.divUp2.text = "" + Math.round((minMinVal + (maxMaxVal -
    minMinVal)*2/7)*1000)/1000;
                this.divUp3.text = "" + Math.round((minMinVal + (maxMaxVal -
    minMinVal)*3/7)*1000)/1000;
                this.divUp4.text = "" + Math.round((minMinVal + (maxMaxVal -
    minMinVal)*4/7)*1000)/1000;
                this.divUp5.text = "" + Math.round((minMinVal + (maxMaxVal -
    minMinVal)*5/7)*1000)/1000;
                this.divUp6.text = "" + Math.round((minMinVal + (maxMaxVal -
    minMinVal)*6/7)*1000)/1000;
                this.divUp7.text = "" + Math.round(maxMaxVal*1000)/1000;
            }

            if (maxValsE.length > 0 && scale_type == 2 && pogr_type <= 2) {
                this.divUp0err.text = "" + Math.round(minValsE[s_cur]*1000)/1000;
                this.divUp1err.text = "" + Math.round((minValsE[s_cur] + (maxValsE[s_cur] -
    minValsE[s_cur])*1/7)*1000)/1000;
                this.divUp2err.text = "" + Math.round((minValsE[s_cur] + (maxValsE[s_cur] -
    minValsE[s_cur])*2/7)*1000)/1000;
                this.divUp3err.text = "" + Math.round((minValsE[s_cur] + (maxValsE[s_cur] -
    minValsE[s_cur])*3/7)*1000)/1000;
                this.divUp4err.text = "" + Math.round((minValsE[s_cur] + (maxValsE[s_cur] -
    minValsE[s_cur])*4/7)*1000)/1000;
                this.divUp5err.text = "" + Math.round((minValsE[s_cur] + (maxValsE[s_cur] -
    minValsE[s_cur])*5/7)*1000)/1000;
```

```
427.                this.divUp6err.text = "" + Math.round((minValsE[s_cur] + (maxValsE[s_cur] -
    minValsE[s_cur])*6/7)*1000)/1000;
428.                this.divUp7err.text = "" + Math.round(maxValsE[s_cur]*1000)/1000;
429.            }
430.            else if (pogr_type <= 2) {
431.                this.divUp0err.text = "" + Math.round(minMinValE*1000)/1000;
432.                this.divUp1err.text = "" + Math.round((minMinValE + (maxMaxValE -
    minMinValE)*1/7)*1000)/1000;
433.                this.divUp2err.text = "" + Math.round((minMinValE + (maxMaxValE -
    minMinValE)*2/7)*1000)/1000;
434.                this.divUp3err.text = "" + Math.round((minMinValE + (maxMaxValE -
    minMinValE)*3/7)*1000)/1000;
435.                this.divUp4err.text = "" + Math.round((minMinValE + (maxMaxValE -
    minMinValE)*4/7)*1000)/1000;
436.                this.divUp5err.text = "" + Math.round((minMinValE + (maxMaxValE -
    minMinValE)*5/7)*1000)/1000;
437.                this.divUp6err.text = "" + Math.round((minMinValE + (maxMaxValE -
    minMinValE)*6/7)*1000)/1000;
438.                this.divUp7err.text = "" + Math.round(maxMaxValE*1000)/1000;
439.            }
440.            else {
441.                this.divUp0err.text = "" + Math.round(minEotT*1000)/1000;
442.                this.divUp1err.text = "" + Math.round((minEotT + (maxEotT - minEotT)*1/7)*1000)/1000;
443.                this.divUp2err.text = "" + Math.round((minEotT + (maxEotT - minEotT)*2/7)*1000)/1000;
444.                this.divUp3err.text = "" + Math.round((minEotT + (maxEotT - minEotT)*3/7)*1000)/1000;
445.                this.divUp4err.text = "" + Math.round((minEotT + (maxEotT - minEotT)*4/7)*1000)/1000;
446.                this.divUp5err.text = "" + Math.round((minEotT + (maxEotT - minEotT)*5/7)*1000)/1000;
447.                this.divUp6err.text = "" + Math.round((minEotT + (maxEotT - minEotT)*6/7)*1000)/1000;
448.                this.divUp7err.text = "" + Math.round(maxEotT*1000)/1000;
449.            }
450.
451.            t_cur = s_cur*t;
452.
453.            this.step_text.text = "t = " + Math.round(t_cur*10000)/10000 + " (шаг " +
    Math.round(s_cur) + ")";
454.
455.            this.axisDown.y = Math.min(Math.max(transfY(0), 26.4), 418.6);
456.
457.            this.axisDownErr.y = Math.min(Math.max(transfEY(0), 468), 700);
458.        }
459.
460.
461.        //baseGraph
462.        //x = 0 : posX = 60
463.        //y = 0 : posY = 418.6
464.
465.        //x = max : posX = 627
466.        //y = max : posY = 26.4
467.
468.        //errGraph
469.        //x = 0 : posX = 53.6
470.        //y = 0 : posY = 700
471.
472.        //x = max : posX = 397.8
473.        //y = max : posY = 468.4
474.
475.
476.
477.
478.
479.        var t_cur = 0;
480.        //текущий момент времени
481.
482.        var s_cur = 0;
483.        //текущий шаг
484.
485.
486.        var etalon = [];
```

```
487.        var xVector = [];
488.        var tVector = [];
489.
490.        var maxMaxVal = 1;
491.        var minMinVal = 0;
492.
493.        var maxMaxValE = 1; //link
494.        var minMinValE = 0; //link
495.        var maxMaxValE1 = 1;
496.        var minMinValE1 = 0;
497.        var maxMaxValE2 = 1;
498.        var minMinValE2 = 0;
499.
500.        var maxVals = [];
501.        var minVals = [];
502.
503.        var maxValsE = []; //link
504.        var minValsE = []; //link
505.        var maxValsE1 = [];
506.        var minValsE1 = [];
507.        var maxValsE2 = [];
508.        var minValsE2 = [];
509.
510.
511.        var EotT = [];
512.
513.        this.addEventListener("tick", setError.bind(this));
514.        function setError() {
515.            if (pogr_type == 1) {
516.                maxMaxValE = maxMaxValE1;
517.                minMinValE = minMinValE1;
518.                maxValsE = maxValsE1;
519.                minValsE = minValsE1;
520.                errorGraph = absolute_error;
521.            }
522.            else {
523.                maxMaxValE = maxMaxValE2;
524.                minMinValE = minMinValE2;
525.                maxValsE = maxValsE2;
526.                minValsE = minValsE2;
527.                errorGraph = relative_error;
528.            }
529.        }
530.
531.        var solve = [];
532.
533.
534.        function copyV(vector) {
535.            var newV = [];
536.            for (var i = 0 ; i < vector.length ; ++i) {
537.                newV.push(vector[i]);
538.            }
539.            return newV;
540.        }
541.
542.
543.        var absolute_error = [];
544.        var relative_error = [];
545.        var errorGraph = [];
546.
547.        var minEotT = 0;
548.        var maxEotT = 1;
549.
550.
551.
552.        function explicit_step(i, vector0, vector1) {
553.            var curT = i * t;
554.
```

```
555.            //Сначала вычисляем серединку
556.            for (var j = 1 ; j < N ; ++j) {
557.                vector1[j] = sigma * vector0[j + 1] + (1 - 2*sigma) * vector0[j] + sigma * vector0[j -
    1] + 0.5*Math.exp(-0.5*curT)*Math.sin((j-1)*h)*t;
558.
559.                maxVals[i] = (vector1[j] > maxVals[i] ? vector1[j] : maxVals[i]);
560.                minVals[i] = (vector1[j] < minVals[i] ? vector1[j] : minVals[i]);
561.            }
562.
563.            //аппроксимируем граничные условия
564.            if (meth_type == 1) {
565.                vector1[0] = vector1[1] - (h/alpha)*phi_0(curT);
566.                vector1[N] = vector1[N-1] + (h/gamma)*phi_l(curT);
567.            }
568.            if (meth_type == 2) {
569.                //(u1k - u0k)/h = (-3u0k + 4u1k - u2k)/2h
570.                //бывожу u0k
571.                //получаю u0k = 2u1k - u2k
572.                //аналогично для ulk
573.                vector1[0] = (4 * vector1[1] - vector1[2] - 2*h*phi_0(curT))/3;
574.                vector1[N] = (4 * vector1[N-1] - vector1[N-2] + 2*h*phi_l(curT))/3;
575.            }
576.            if (meth_type == 3) {
577.                //тут идут сложные преобразования, которые
578.                //я делал на бумаге
579.                var d0 = h*vector0[0]/t - (phi_0(curT)*(2*a*a/alpha));
580.                var c0 = -2 * a * a / h;
581.                var b0 = (2 * a * a / h) + (h/t);
582.                var a0 = 0;
583.
584.                vector1[0] = (d0 - c0*vector1[1]) / b0;
585.
586.                var dn = h*vector0[N]/t + (phi_l(curT)*(2*a*a/gamma));
587.                var cn = 0;
588.                var bn = 2*a*a/h + (h/t);
589.                var an = -2*a*a/h;
590.
591.                vector1[N] = (dn - an*vector1[N-1])/bn;
592.            }
593.
594.            maxVals[i] = (vector1[0] > maxVals[i] ? vector1[0] : maxVals[i]);
595.            maxVals[i] = (vector1[N] > maxVals[i] ? vector1[N] : maxVals[i]);
596.
597.            minVals[i] = (vector1[0] < minVals[i] ? vector1[0] : minVals[i]);
598.            minVals[i] = (vector1[N] < minVals[i] ? vector1[N] : minVals[i]);
599.
600.            maxMaxVal = (maxVals[i] > maxMaxVal ? maxVals[i] : maxMaxVal);
601.            minMinVal = (minVals[i] < minMinVal ? minVals[i] : minMinVal);
602.
603.            return copyV(vector1);
604.
605.        }
606.
607.
608.
609.        function implicit_step(i, vector0, vector1) {
610.            var curT = i * t;
611.
612.            var solveMatrix = [];
613.            var solveVector = [];
614.
615.            var emptV = [];
616.            for (var j = 0 ; j <= N ; ++j) {
617.                emptV.push(0);
618.            }
619.
620.            var thisV = copyV(emptV);
621.
```

```
622.            if (meth_type == 1) {
623.                thisV[0] = -1; //b0
624.                thisV[1] = 1; //c0
625.
626.                solveMatrix.push(copyV(thisV));
627.
628.                var d0 = h*phi_0(curT);
629.
630.                solveVector.push(d0); //d0
631.            }
632.            if (meth_type == 2) {
633.                thisV[0] = -3;
634.                thisV[1] = 4;
635.                thisV[2] = -1
636.
637.                solveMatrix.push(copyV(thisV));
638.
639.                var d0 = phi_0(curT) * 2 * h;
640.
641.                solveVector.push(d0); //d0
642.            }
643.            if (meth_type == 3) {
644.                thisV[0] = (2*a*a / h) + (h/t); //b0 g(c) u beta = 0
645.                thisV[1] = (-2*a*a / h);
646.
647.                solveMatrix.push(copyV(thisV));
648.
649.                var d0 = h/t * vector0[0] - phi_0(curT)*(2*a*a / alpha); //b = 0
650.
651.                solveVector.push(d0); //d0
652.            }
653.
654.            for (var j = 1 ; j < N ; ++j) {
655.                var thisV = copyV(emptV);
656.                thisV[j - 1] = -sigma; //aj
657.                thisV[j] = (2*sigma + 1); //bj
658.                thisV[j + 1] = -sigma; //cj
659.
660.                solveMatrix.push(copyV(thisV));
661.
662.                var dj = vector0[j] + 0.5*Math.exp(-0.5*curT)*Math.sin((j-1)*h)*t;
663.                solveVector.push(dj);
664.
665.                //убираем 3 элемент (для трехдиагональности)
666.                if (j == 1 && meth_type == 2) {
667.                    var divisor = solveMatrix[0][2] / solveMatrix[1][2];
668.                    solveMatrix[0][0] -= solveMatrix[1][0] * divisor;
669.                    solveMatrix[0][1] -= solveMatrix[1][1] * divisor;
670.                    solveMatrix[0][2] = 0;
671.
672.                    solveVector[0] -= solveVector[1] * divisor;
673.                }
674.
675.
676.
677.            }
678.
679.            var thisV = copyV(emptV);
680.
681.            if (meth_type == 1) {
682.                thisV[N-1] = -1; //an
683.                thisV[N] = 1; //bn
684.
685.                solveMatrix.push(copyV(thisV));
686.
687.                var dn = h*phi_1(curT);
688.
689.                solveVector.push(dn); //d1
```

```
690.            }
691.            if (meth_type == 2) {
692.                thisV[N-2] = 1;
693.                thisV[N-1] = -4;
694.                thisV[N] = 3;
695.
696.                solveMatrix.push(copyV(thisV));
697.
698.                var d0 = 2*h*phi_l(curT);
699.
700.                solveVector.push(d0); //d0
701.
702.                var divisor = solveMatrix[N][N-2] / solveMatrix[N-1][N-2];
703.                solveMatrix[N][N] -= solveMatrix[N-1][N] * divisor;
704.                solveMatrix[N][N-1] -= solveMatrix[N-1][N-1] * divisor;
705.                solveMatrix[N][N-2] = 0;
706.
707.                solveVector[N] -= solveVector[N-1] * divisor;
708.            }
709.            if (meth_type == 3) {
710.                thisV[N-1] = 2*a*a/h; //an
711.                thisV[N] = (2*a*a / h) + (h/t);  //bn g(c) u delta = 0
712.
713.                solveMatrix.push(copyV(thisV));
714.
715.                var dn = h/t * vector0[N] - phi_l(curT)*(2*a*a / gamma); //b = 0
716.
717.                solveVector.push(dn); //d1
718.            }
719.
720.            var the_vector = progonka(solveMatrix, solveVector);
721.
722.
723.            for (var j = 0 ; j <= N ; ++j) {
724.
725.                maxVals[i] = (the_vector[j] > maxVals[i] ? the_vector[j] : maxVals[i]);
726.                minVals[i] = (the_vector[j] < minVals[i] ? the_vector[j] : minVals[i]);
727.
728.                maxMaxVal = (maxVals[i] > maxMaxVal ? maxVals[i] : maxMaxVal);
729.                minMinVal = (minVals[i] < minMinVal ? minVals[i] : minMinVal);
730.
731.            }
732.
733.            return copyV(the_vector);
734.        }
735.
736.
737.
738.        this.beginBtn.addEventListener("click", beginSimulation.bind(this));
739.        function beginSimulation() {
740.            generation += 1;
741.            s_cur = 0;
742.            etalon = [];
743.            solve = [];
744.            maxMaxVal = 1;
745.            minMinVal = 0;
746.
747.            maxVals = [];
748.            minVals = [];
749.
750.            xVector = [];
751.            tVector = [];
752.
753.            for (var j = 0 ; j <= N ; ++j) {
754.                xVector.push(h * j);
755.            }
756.            for (var i = 0 ; i <= K ; ++i) {
757.                var timeVector = [];
```

```
758.                 var this_t = t * i;
759.                 tVector.push(this_t);
760.                 var this_max = 0;
761.                 var this_min = 0;
762.                 for (var j = 0 ; j <= N ; ++j) {
763.                     var this_x = xVector[j];
764.                     var func_res = U(this_x, this_t);
765.                     if (j == 0) {
766.                         this_max = func_res;
767.                         this_min = func_res;
768.                     }
769.                     else {
770.                         this_max = (func_res > this_max ? func_res : this_max);
771.                         this_min = (func_res < this_min ? func_res : this_min);
772.                     }
773.                     timeVector.push(func_res);
774.                 }
775.                 etalon.push(timeVector);
776.                 maxVals.push(this_max);
777.                 minVals.push(this_min);
778.                 if (i == 0) {
779.                     maxMaxVal = this_max;
780.                     minMinVal = this_min;
781.                 }
782.                 else {
783.                     maxMaxVal = (this_max > maxMaxVal ? this_max : maxMaxVal);
784.                     minMinVal = (this_min < minMinVal ? this_min : minMinVal);
785.                 }
786.             }
787.             //задаем начальное условие
788.             var psiGrid0 = [];
789.             var emptV = [];
790.             for (var j = 0 ; j <= N ; ++j) {
791.                 psiGrid0.push(psi(xVector[j]));
792.                 emptV.push(0);
793.             }
794.             solve.push(psiGrid0);
795.
796.             var vector0 = psiGrid0;
797.             var vector1 = copyV(emptV);
798.
799.             //scheme 1
800.             if (scheme_type == 1) {
801.
802.                 for (var i = 1 ; i <= K ; ++i) {
803.
804.                     var resV = explicit_step(i, vector0, vector1);
805.
806.                     solve.push(copyV(resV));
807.
808.                     vector0 = copyV(resV);
809.                     vector1 = copyV(emptV);
810.
811.                 }
812.
813.             }
814.             else if (scheme_type == 2) {
815.
816.                 for (var i = 1 ; i <= K ; ++i) {
817.
818.                     var resV = implicit_step(i, vector0, vector1);
819.
820.                     solve.push(copyV(resV));
821.
822.                     vector0 = copyV(resV);
823.                     vector1 = copyV(emptV);
824.
825.                 }
```

```
826.            }
827.
828.            else if (scheme_type == 3) {
829.
830.                for (var i = 1 ; i <= K ; ++i) {
831.
832.                    var resV1 = explicit_step(i, vector0, vector1);
833.                    var resV2 = implicit_step(i, vector0, vector1);
834.
835.                    var totalV = [];
836.
837.                    for (var y = 0 ; y < resV1.length ; ++y) {
838.                        var resT = 0.5*resV1[y] + 0.5*resV2[y];
839.                        totalV.push(resT);
840.                    }
841.
842.                    solve.push(copyV(totalV));
843.
844.                    vector0 = copyV(totalV);
845.                    vector1 = copyV(emptV);
846.
847.                }
848.            }
849.
850.
851.            // error
852.
853.
854.            absolute_error = [];
855.            relative_error = [];
856.            maxMaxValE1 = 1;
857.            minMinValE1 = 0;
858.            maxMaxValE2 = 1;
859.            minMinValE2 = 0;
860.
861.            maxValsE1 = [];
862.            minValsE1 = [];
863.            maxValsE2 = [];
864.            minValsE2 = [];
865.
866.            EotT = [];
867.
868.            for (var i = 0 ; i <= K ; ++i) {
869.                var timeVector1 = [];
870.                var timeVector2 = [];
871.                var this_t = t * i;
872.                var this_max1 = 0;
873.                var this_min1 = 0;
874.
875.                var this_max2 = 0;
876.                var this_min2 = 0;
877.
878.                for (var j = 0 ; j <= N ; ++j) {
879.                    var this_x = xVector[j];
880.
881.                    var err1 = solve[i][j] - etalon[i][j];
882.                    var err2 = 0;
883.                    if (Math.abs(etalon[i][j]) > 0.1) {
884.                        err2 = Math.abs(1 - solve[i][j]/etalon[i][j]);
885.                    }
886.                    else {
887.                        err2 = Math.abs(1 - (solve[i][j]+0.2)/(etalon[i][j] + 0.2));
888.                    }
889.
890.                    if (j == 0) {
891.                        this_max1 = err1;
892.                        this_min1 = err1;
893.
```

```
                        this_max2 = err2;
                        this_min2 = err2;
                    }
                    else {
                        this_max1 = (err1 > this_max1 ? err1 : this_max1);
                        this_min1 = (err1 < this_min1 ? err1 : this_min1);

                        this_max2 = (err2 > this_max2 ? err2 : this_max2);
                        this_min2 = (err2 < this_min2 ? err2 : this_min2);
                    }
                    timeVector1.push(err1);
                    timeVector2.push(err2);
                }

                absolute_error.push(timeVector1);
                relative_error.push(timeVector2);

                maxValsE1.push(this_max1);
                minValsE1.push(this_min1);

                maxValsE2.push(this_max2);
                minValsE2.push(this_min2);


                if (i == 0) {
                    maxMaxValE1 = this_max1;
                    minMinValE1 = this_min1;

                    maxMaxValE2 = this_max2;
                    minMinValE2 = this_min2;

                    var stepEot = Math.max(maxMaxValE1, Math.abs(minMinValE1));

                    EotT.push(stepEot);

                    minEotT = stepEot;
                    maxEotT = stepEot;
                }
                else {
                    maxMaxValE1 = (this_max1 > maxMaxValE1 ? this_max1 : maxMaxValE1);
                    minMinValE1 = (this_min1 < minMinValE1 ? this_min1 : minMinValE1);

                    var stepEot = Math.max(this_max1, Math.abs(this_min1));

                    minEotT = (stepEot < minEotT ? stepEot : minEotT);
                    maxEotT = (stepEot > maxEotT ? stepEot : maxEotT);

                    EotT.push(stepEot);



                    maxMaxValE2 = (this_max2 > maxMaxValE2 ? this_max2 : maxMaxValE2);
                    minMinValE2 = (this_min2 < minMinValE2 ? this_min2 : minMinValE2);
                }
            }



            makeEtalonGraph();
            makeSolveGraph();
            makeErrorGraph();
            makeErrorTGraph();

            go = true;
        }

        function transfX(x) {
            var newX = 60 + (567 * (x / 1));
```

```
962.            return newX;
963.        }
964.
965.        function transfY(y) {
966.            var newY = 0;
967.            if (maxVals.length > 0 && scale_type == 2) {
968.                newY = 418.6 - (382.6 * (y - minVals[s_cur])/(maxVals[s_cur] - minVals[s_cur]));
969.            }
970.            else {
971.                newY = 418.6 - (382.6 * (y - minMinVal)/(maxMaxVal - minMinVal));
972.            }
973.            return newY;
974.        }
975.
976.
977.        function transfEX(x) {
978.            var newX = 60.6 + (344.2 * (x / l));
979.            return newX;
980.        }
981.
982.        function transfE2X(x) {
983.            var newX = 60.6 + (344.2 * (x / T));
984.            return newX;
985.        }
986.
987.        function transfEY(y) {
988.            var newY = 0;
989.            if (maxValsE.length > 0 && scale_type == 2) {
990.                newY = 700 - (231 * (y - minValsE[s_cur])/(maxValsE[s_cur] - minValsE[s_cur]));
991.            }
992.            else {
993.                newY = 700 - (231 * (y - minMinValE)/(maxMaxValE - minMinValE));
994.            }
995.            return newY;
996.        }
997.
998.        function transfE2Y(y) {
999.            var newY = 0;
1000.           newY = 700 - (231 * (y - minEotT)/(maxEotT - minEotT));
1001.
1002.           return newY;
1003.       }
1004.
1005.
1006.       function makeEtalonGraph() {
1007.           for (var i = 0 ; i < N ; ++i) {
1008.               var join = new lib.line();
1009.               stage.addChild(join);
1010.               join.x = transfX(xVector[i]);
1011.               join.y = transfY(etalon[0][i]);
1012.               join.endX = transfX(xVector[i+1]);
1013.               join.endY = transfY(etalon[0][i+1]);
1014.
1015.               join.gotoAndStop(0);
1016.               join.num = i;
1017.
1018.               join.len = Math.sqrt(Math.pow((join.endY - join.y), 2) + Math.pow((join.endX -
    join.x), 2));
1019.
1020.               join.scaleX = join.len;
1021.               join.scaleY = 1;
1022.
1023.               join.rotation = Math.atan2((join.endY - join.y), (join.endX - join.x)) * 180 /
    Math.PI;
1024.
1025.               join.gen = generation;
1026.
1027.
```

```
1028.                 join.visible = true;
1029.                 join.alpha = 1;
1030.
1031.                 join.addEventListener('tick', setPoses);
1032.             }
1033.
1034.         }
1035.
1036.         function makeSolveGraph() {
1037.             for (var i = 0 ; i < N ; ++i) {
1038.                 var join = new lib.line();
1039.                 stage.addChild(join);
1040.                 join.x = transfX(xVector[i]);
1041.                 join.y = transfY(solve[0][i]);
1042.                 join.endX = transfX(xVector[i+1]);
1043.                 join.endY = transfY(solve[0][i+1]);
1044.
1045.                 join.gotoAndStop(1);
1046.                 join.num = i;
1047.
1048.                 join.len = Math.sqrt(Math.pow((join.endY - join.y), 2) + Math.pow((join.endX -
    join.x), 2));
1049.
1050.                 join.scaleX = join.len;
1051.                 join.scaleY = 1;
1052.
1053.                 join.rotation = Math.atan2((join.endY - join.y), (join.endX - join.x)) * 180 /
    Math.PI;
1054.
1055.                 join.gen = generation;
1056.
1057.
1058.                 join.visible = true;
1059.                 join.alpha = 1;
1060.
1061.                 join.addEventListener('tick', setPoses2);
1062.             }
1063.
1064.         }
1065.
1066.         function makeErrorGraph() {
1067.             setError();
1068.             for (var i = 0 ; i < N ; ++i) {
1069.                 var join = new lib.line();
1070.                 stage.addChild(join);
1071.                 join.x = transfEX(xVector[i]);
1072.                 join.y = transfEY(errorGraph[0][i]);
1073.                 join.endX = transfEX(xVector[i+1]);
1074.                 join.endY = transfEY(errorGraph[0][i+1]);
1075.
1076.                 join.gotoAndStop(3);
1077.                 join.num = i;
1078.
1079.                 join.len = Math.sqrt(Math.pow((join.endY - join.y), 2) + Math.pow((join.endX -
    join.x), 2));
1080.
1081.                 join.scaleX = join.len;
1082.                 join.scaleY = 1;
1083.
1084.                 join.rotation = Math.atan2((join.endY - join.y), (join.endX - join.x)) * 180 /
    Math.PI;
1085.
1086.                 join.gen = generation;
1087.
1088.
1089.                 join.visible = true;
1090.                 join.alpha = 1;
1091.
```

```
1092.                 join.addEventListener('tick', setPoses3);
1093.             }
1094.
1095.         }
1096.
1097.         function makeErrorTGraph() {
1098.             for (var i = 0 ; i < K ; ++i) {
1099.                 var join = new lib.line();
1100.                 stage.addChild(join);
1101.                 join.x = transfE2X(tVector[i]);
1102.                 join.y = transfE2Y(EotT[i]);
1103.                 join.endX = transfE2X(tVector[i+1]);
1104.                 join.endY = transfE2Y(tVector[i+1]);
1105.
1106.                 join.gotoAndStop(3);
1107.                 join.num = i;
1108.
1109.                 join.len = Math.sqrt(Math.pow((join.endY - join.y), 2) + Math.pow((join.endX -
    join.x), 2));
1110.
1111.                 join.scaleX = join.len;
1112.                 join.scaleY = 1;
1113.
1114.                 join.rotation = Math.atan2((join.endY - join.y), (join.endX - join.x)) * 180 /
    Math.PI;
1115.
1116.                 join.gen = generation;
1117.
1118.
1119.                 join.visible = true;
1120.                 join.alpha = 1;
1121.
1122.                 join.addEventListener('tick', setPoses4);
1123.             }
1124.
1125.         }
1126.
1127.         function setPoses(e) {
1128.             var object = e.currentTarget;
1129.             if (object.gen == generation) {
1130.                 object.x = transfX(xVector[object.num]);
1131.                 object.y = transfY(etalon[s_cur][object.num]);
1132.                 object.endX = transfX(xVector[object.num+1]);
1133.                 object.endY = transfY(etalon[s_cur][object.num+1]);
1134.
1135.                 object.len = Math.sqrt(Math.pow((object.endY - object.y), 2) + Math.pow((object.endX -
    object.x), 2));
1136.
1137.                 object.scaleX = object.len;
1138.                 object.scaleY = 1;
1139.
1140.                 object.rotation = Math.atan2((object.endY - object.y), (object.endX - object.x)) * 180
    / Math.PI;
1141.             }
1142.             else if (object.gen != generation) {
1143.                 object.alpha -= 3/30;
1144.             }
1145.
1146.             if (object.alpha <= 0) {
1147.                 object.alpha = 0;
1148.                 object.visible = false;
1149.                 object.removeEventListener('tick', setPoses);
1150.                 stage.removeChild(object);
1151.             }
1152.
1153.         }
1154.
1155.
```

```
1156.       function setPoses2(e) {
1157.           var object = e.currentTarget;
1158.           if (object.gen == generation) {
1159.               object.x = transfX(xVector[object.num]);
1160.               object.y = transfY(solve[s_cur][object.num]);
1161.               object.endX = transfX(xVector[object.num+1]);
1162.               object.endY = transfY(solve[s_cur][object.num+1]);
1163.
1164.               object.len = Math.sqrt(Math.pow((object.endY - object.y), 2) + Math.pow((object.endX -
    object.x), 2));
1165.
1166.               object.scaleX = object.len;
1167.               object.scaleY = 1;
1168.
1169.               object.rotation = Math.atan2((object.endY - object.y), (object.endX - object.x)) * 180
    / Math.PI;
1170.           }
1171.           else if (object.gen != generation) {
1172.               object.alpha -= 3/30;
1173.           }
1174.
1175.           if (object.alpha <= 0) {
1176.               object.alpha = 0;
1177.               object.visible = false;
1178.               object.removeEventListener('tick', setPoses2);
1179.               stage.removeChild(object);
1180.           }
1181.
1182.       }
1183.
1184.
1185.       function setPoses3(e) {
1186.           var object = e.currentTarget;
1187.           if (object.gen == generation) {
1188.               object.x = transfEX(xVector[object.num]);
1189.               object.y = transfEY(errorGraph[s_cur][object.num]);
1190.               object.endX = transfEX(xVector[object.num+1]);
1191.               object.endY = transfEY(errorGraph[s_cur][object.num+1]);
1192.
1193.               object.len = Math.sqrt(Math.pow((object.endY - object.y), 2) + Math.pow((object.endX -
    object.x), 2));
1194.
1195.               object.scaleX = object.len;
1196.               object.scaleY = 1;
1197.
1198.               object.rotation = Math.atan2((object.endY - object.y), (object.endX - object.x)) * 180
    / Math.PI;
1199.           }
1200.           else if (object.gen != generation) {
1201.               object.alpha -= 3/30;
1202.           }
1203.
1204.           if (pogr_type <= 2) {
1205.               object.visible = true;
1206.           }
1207.           else {
1208.               object.visible = false;
1209.           }
1210.
1211.           if (object.alpha <= 0) {
1212.               object.alpha = 0;
1213.               object.visible = false;
1214.               object.removeEventListener('tick', setPoses3);
1215.               stage.removeChild(object);
1216.           }
1217.
1218.       }
1219.
```

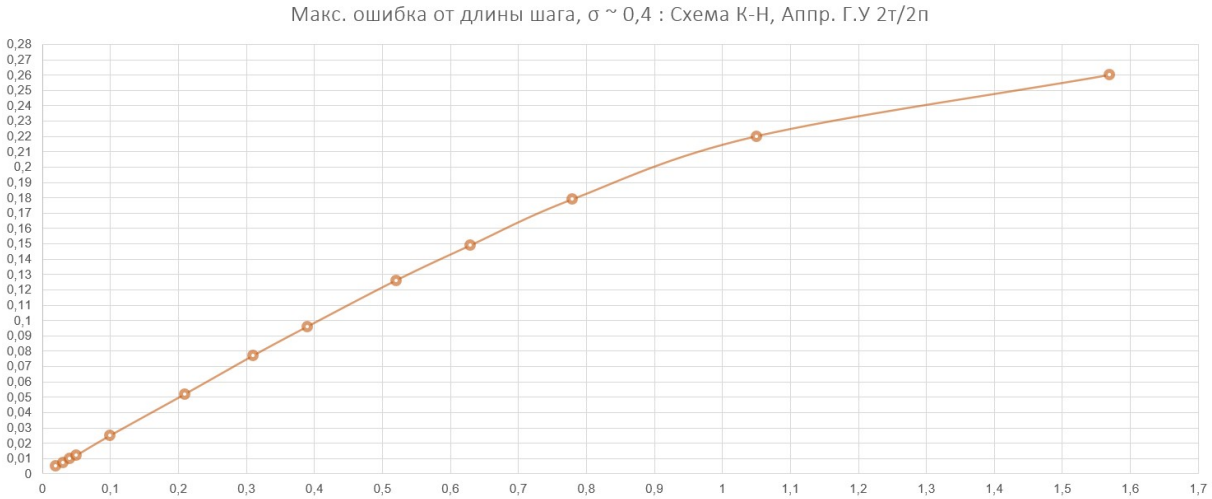
```
1220.
1221.       function setPoses4(e) {
1222.           var object = e.currentTarget;
1223.           if (object.gen == generation) {
1224.               object.x = transfE2X(tVector[object.num]);
1225.               object.y = transfE2Y(EotT[object.num]);
1226.               object.endX = transfE2X(tVector[object.num+1]);
1227.               object.endY = transfE2Y(EotT[object.num+1]);
1228.
1229.               object.len = Math.sqrt(Math.pow((object.endY - object.y), 2) + Math.pow((object.endX -
    object.x), 2));
1230.
1231.               object.scaleX = object.len;
1232.               object.scaleY = 1;
1233.
1234.               object.rotation = Math.atan2((object.endY - object.y), (object.endX - object.x)) * 180
    / Math.PI;
1235.           }
1236.           else if (object.gen != generation) {
1237.               object.alpha -= 3/30;
1238.           }
1239.
1240.           if (pogr_type == 3) {
1241.               object.visible = true;
1242.           }
1243.           else {
1244.               object.visible = false;
1245.           }
1246.
1247.           if (object.alpha <= 0) {
1248.               object.alpha = 0;
1249.               object.visible = false;
1250.               object.removeEventListener('tick', setPoses4);
1251.               stage.removeChild(object);
1252.           }
1253.
1254.       }
1255.
1256.
1257.
1258.
1259.
1260.
1261.
1262.
1263.       //метод прогонки
1264.       function progonka(matrix, vectorB) {
1265.           var vectorX = [];
1266.
1267.           var N = vectorB.length;
1268.
1269.           var alphas = [];
1270.           var betas = [];
1271.
1272.
1273.           for (var i = 0 ; i < vectorB.length ; ++i) {
1274.               alphas.push(0);
1275.               betas.push(0);
1276.           }
1277.
1278.           //Прямой ход прогонки
1279.
1280.           for (var i = 0; i < N; ++i) {
1281.               var A0, C0, B0, F0;
1282.
1283.               if (i - 1 < 0) {
1284.                   A0 = 0;
1285.               }
```

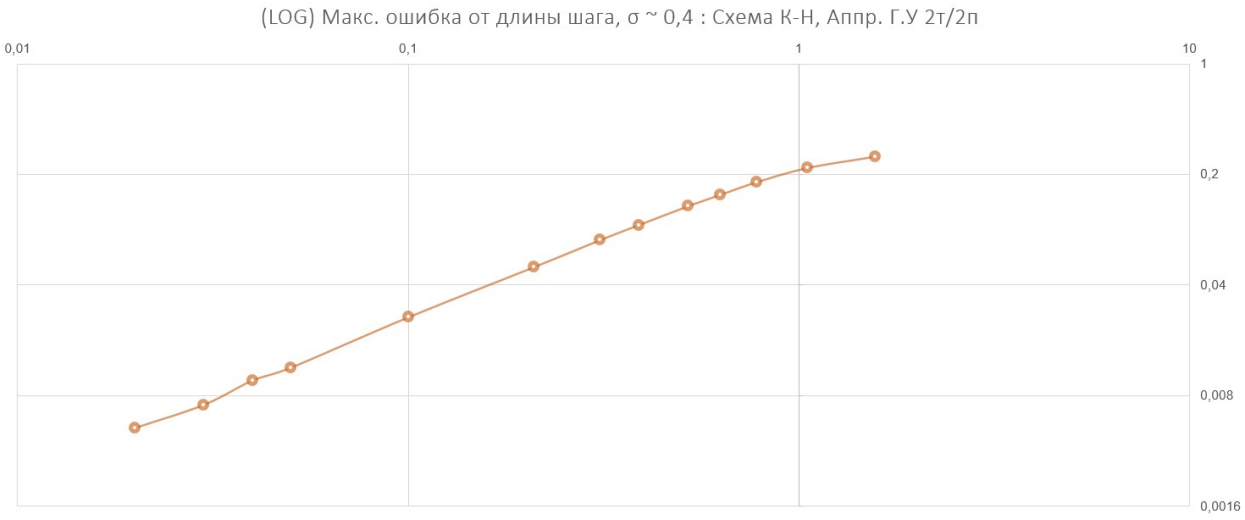
```
1286.                else {
1287.                    A0 = matrix[i][i - 1];
1288.                }
1289.
1290.                C0 = -1 * matrix[i][i];
1291.
1292.                if (i + 1 < N) {
1293.                    B0 = matrix[i][i + 1];
1294.                }
1295.                else {
1296.                    B0 = 0;
1297.                }
1298.
1299.                F0 = vectorB[i];
1300.
1301.
1302.                if (i == 0) {
1303.                    alphas[i] = B0 / C0;
1304.                    betas[i] = -(F0 / C0);
1305.                }
1306.                else if (i == N - 1) {
1307.                    alphas[i] = 0;
1308.                    betas[i] = (betas[i - 1] * A0 - F0) / (C0 - alphas[i - 1] * A0);
1309.                }
1310.                else {
1311.                    alphas[i] = B0 / (C0 - alphas[i - 1] * A0);
1312.                    betas[i] = (betas[i - 1] * A0 - F0) / (C0 - alphas[i - 1] * A0);
1313.                }
1314.
1315.            }
1316.
1317.
1318.            vectorX[N - 1] = betas[N - 1];
1319.
1320.            for (var i = 2; i <= N; ++i) {
1321.                vectorX[N - i] = alphas[N - i] * vectorX[N - i + 1] + betas[N - i];
1322.            }
1323.
1324.            return vectorX;
1325.        }
```

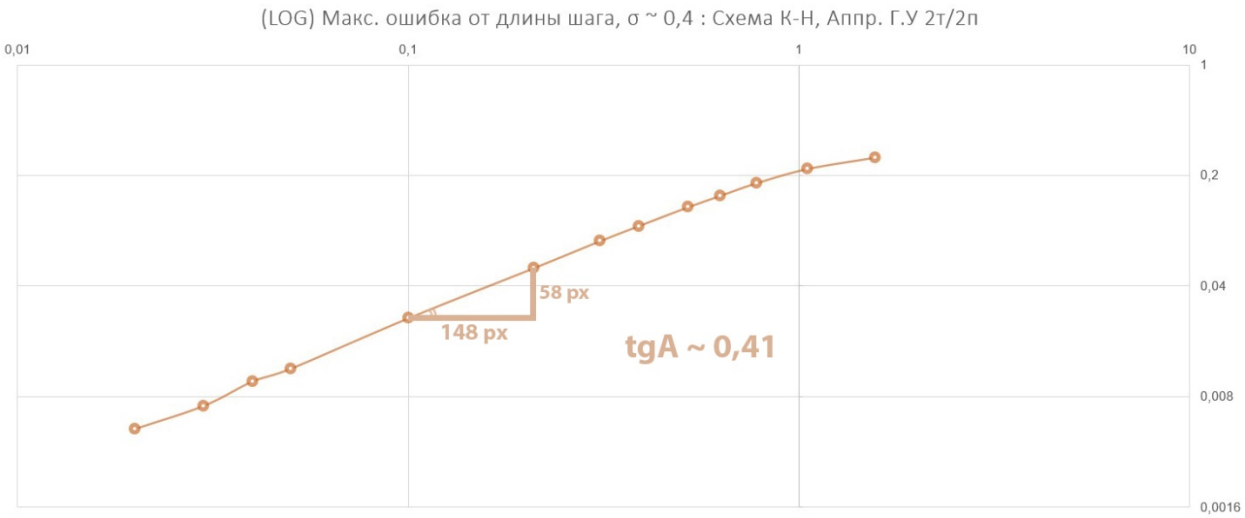
7. Графики

Ошибка от времени при T = 10

Ошибка от размера шага – обычный и логарифмические графики







8. Данная лабораторная работа выполнена: **17 января 2021.**

Лабораторная работа 6

1. Тема ЛР:

   Метод конечных разностей для решения уравнений гиперболического типа. Явная крест-схема и неявная.

2. Вариант : **7**

7.

$$\frac{\partial^2 u}{\partial t^2} + 2\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + 2\frac{\partial u}{\partial x} - 3u,$$

$$u(0,t) = \exp(-t)\cos(2t),$$

$$u(\frac{\pi}{2}, t) = 0,$$

$$u(x,0) = \exp(-x)\cos x,$$

$$u_t(x,0) = -\exp(-x)\cos x.$$

Аналитическое решение: $U(x,t) = \exp(-t-x)\cos x\cos(2t)$

3. Алгоритм:

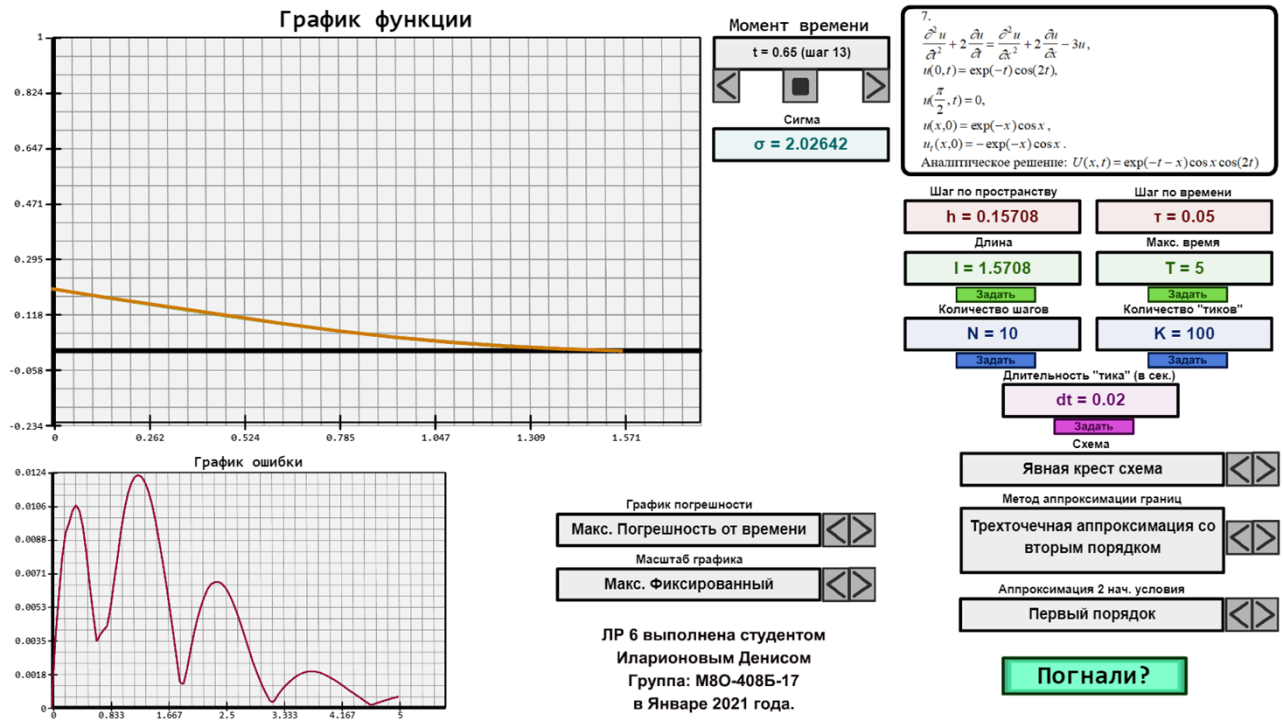
   После предыдущей лабораторной, эта мне показалась куда легче. Используются похожие схемы, однако, теперь, я знаю, как решать такие уравнения. Явная крест схема использует значения двух предыдущих временных слоев. Поэтому, здесь было кое-что необычное – аппроксимация второго начального условия. Именно так мы находим значения во втором временном слое. К счастью, в моем варианте границы были заданы явно, так что, не пришлось долго возиться с этим.

4. Среда разработки:

Adobe Animate, Javascript + HTML5

5. Реализация

Можно указать порядок аппроксимации 2 начального условия. Выбрать схему. Посмотреть погрешность, а также все то, что было в предыдущей лабораторной.



6. Код программы

```
1.  var alpha = 0; //α
2.  var beta = 1; //β
3.  var gamma = 0; //γ
4.  var delta = 1; //δ
5.
6.
7.  //α * Ux(0, t) + β * U(0, t) = exp(-t)cos(2t)
8.  //γ * Ux(l, t) + δ * U(l, t) = 0
9.  //Обозначим все как граничные условия 3 рода
10.
11. function phi_0(t) {
12.     return Math.exp(-1 * t) * Math.cos(2*t);
13. }
14.
15. function phi_l(t) {
16.     return 0;
17. }
18.
19. function psi(x) {
20.     return Math.exp(-x) * Math.cos(x);
21. }
22. // Начальное условие 1 (U(x, 0))
23.
24. function psi2(x) {
25.     return -Math.exp(-x) * Math.cos(x);
26. }
27. // Начальное условие 2 (Ut(x, 0))
```

```
28.
29.
30. function psi2d1(x) {
31.     return Math.exp(-x)*Math.sin(x) + Math.exp(-x)*Math.cos(x);
32. }
33.
34.
35. //Сделать анализ по результатам
36. //Порядок сходимости метода
37. //Зависимость при const числе сигма (куррента)
38. //Макс. Погрешность от шага (в опред. момент времени)
39. //Лучше логарифм
40. //Сделать график зависимости модуля макс. ошибки от времени
41.
42.
43.
44.
45. function psi2d2(x) {
46.     return -2 * Math.exp(-x) * Math.sin(x);
47. }
48. // Первая и вторая производная 2 н.у
49.
50. function U(x, t) {
51.     return Math.exp(-t - x) * Math.cos(x) * Math.cos(2*t);
52. } //Аналитическое решение
53.
54.
55. //Параметры сетки
56. var l = 3.1415926535/2; //интервал
57. var N = 10; //количество шагов
58. var T = 1; //макс. время
59. var K = 100; //количество тиков
60.
61. var h = 0; var t = 0;
62.
63. var a = 1; var b = 2; var c = -3;
64.
65.
66. var sigma = 1;
67. //σ = a^2 * τ / h^2
68.
69. var dt = 0.02; //длительность тика
70. var cur_tick = 0;
71.
72.
73. var generation = 0; //номер генерации
74. var go = false; //увеличивать счетчик тиков?
75.
76.
77. this.addEventListener("tick", det_ht.bind(this));
78. function det_ht() {
79.     h = l / N;
80.     t = T / K;
81.
82.     sigma = a * a * t / (h * h);
83.     //определяем шаги по времени и пространству
84.
85.     this.h_text.text = "h = " + Math.round(h * 100000)/100000;
86.     this.tau_text.text = "τ = " + Math.round(t * 100000)/100000;
87.
88.     this.l_text.text = "l = " + Math.round(l * 100000)/100000;
89.     this.T_text.text = "T = " + Math.round(T * 100000)/100000;
90.
91.     this.N_text.text = "N = " + Math.round(N);
92.     this.K_text.text = "K = " + Math.round(K);
93.
94.     this.dt_text.text = "dt = " + Math.round(dt*1000)/1000;
95.
```

```
96.     this.sig_text.text = "σ = " + Math.round(sigma * 100000)/100000;
97.
98.
99.     if (go) {
100.                cur_tick += 1/60;
101.                if (cur_tick >= dt && s_cur < K) {
102.                    var tGot = Math.floor(cur_tick / dt);
103.                    cur_tick %= dt;
104.                    s_cur += Math.min(K - s_cur, tGot);
105.                }
106.                else if (s_cur == K) {
107.                    go = false;
108.                }
109.            }
110.
111.        }
112.
113.
114.
115.
116.        this.setGreen1.addEventListener("click", set_l_prompt.bind(this));
117.        function set_l_prompt() {
118.            generation += 1;
119.            var temp = prompt("Введите l:", '');
120.            temp = Number.parseFloat(temp);
121.            if (isNaN(temp)) {
122.                //если ввели какую-то хрень
123.                temp = l;
124.            }
125.            else if (temp > 1000000) {
126.                //ограничение на величину
127.                temp = 1000000;
128.            }
129.            else if (temp <= 0) {
130.                //всегда положительно
131.                temp = 0.001;
132.            }
133.            l = temp;
134.        }
135.
136.
137.
138.        this.setGreen2.addEventListener("click", set_T_prompt.bind(this));
139.        function set_T_prompt() {
140.            generation += 1;
141.            var temp = prompt("Введите T:", '');
142.            temp = Number.parseFloat(temp);
143.            if (isNaN(temp)) {
144.                //если ввели какую-то хрень
145.                temp = T;
146.            }
147.            else if (temp > 1000000) {
148.                //ограничение на величину
149.                temp = 1000000;
150.            }
151.            else if (temp <= 0) {
152.                //всегда положительно
153.                temp = 0.001;
154.            }
155.            T = temp;
156.        }
157.
158.
159.        this.setBlue1.addEventListener("click", set_N_prompt.bind(this));
160.        function set_N_prompt() {
161.            generation += 1;
162.            var temp = prompt("Введите N:", '');
163.            temp = Number.parseInt(temp);
```

```
164.            if (isNaN(temp)) {
165.                //если ввели какую-то хрень
166.                temp = N;
167.            }
168.            else if (temp > 200) {
169.                //ограничение на величину
170.                temp = 200;
171.            }
172.            else if (temp <= 0) {
173.                //всегда положительно
174.                temp = 1;
175.            }
176.            N = temp;
177.        }
178.
179.
180.        this.setBlue2.addEventListener("click", set_K_prompt.bind(this));
181.        function set_K_prompt() {
182.            generation += 1;
183.            var temp = prompt("Введите K:", '');
184.            temp = Number.parseInt(temp);
185.            if (isNaN(temp)) {
186.                //если ввели какую-то хрень
187.                temp = K;
188.            }
189.            else if (temp > 10000) {
190.                //ограничение на величину
191.                temp = 10000;
192.            }
193.            else if (temp <= 0) {
194.                //всегда положительно
195.                temp = 1;
196.            }
197.            K = temp;
198.        }
199.
200.
201.        this.setPurp1.addEventListener("click", set_dt_prompt.bind(this));
202.        function set_dt_prompt() {
203.            cur_tick = 0;
204.            var temp = prompt("Введите dt (для ручного переключения можно ввести очень большим):",
    '');
205.            temp = Number.parseFloat(temp);
206.            if (isNaN(temp)) {
207.                //если ввели какую-то хрень
208.                temp = dt;
209.            }
210.            else if (temp > 1000000) {
211.                //ограничение на величину
212.                temp = 1000000;
213.            }
214.            else if (temp < 0.02) {
215.                //всегда положительно и больше 0.2 (почти плавная смена графика)
216.                temp = 0.02;
217.            }
218.            dt = temp;
219.        }
220.
221.
222.
223.        var scheme_type = 1;
224.        //Тип схемы
225.        //1,2 - Явная/Неявная схема
226.
227.        var theta = 1;
228.
229.
230.        var meth_type = 1;
```

```
231.        //Метод аппроксимации г.у
232.        //1 - Двухточечная аппроксимация с первым порядком
233.        //2 - Трехточечная аппроксимация со вторым порядком
234.        //3 - Двухточечная аппроксимация со вторым порядком
235.
236.        var approx_type = 1;
237.        //Метод аппроксимации 2 н.у
238.        //1 - Первый порядок
239.        //2 - Второй порядок
240.
241.
242.
243.        var pogr_type = 1;
244.        //Тип погрешности
245.        //1 - Абсолютная
246.        //2 - Относительная
247.        //3 - От времени
248.
249.        var scale_type = 1;
250.        //Масштаб графика
251.        //1 - Фиксированный
252.        //2 - Динамический
253.
254.        this.SLeft1.addEventListener("click", scheme_left.bind(this));
255.        function scheme_left() {
256.            if (scheme_type == 1) {
257.                scheme_type = 2;
258.            }
259.            else {
260.                scheme_type -= 1;
261.            }
262.        }
263.
264.        this.SRight1.addEventListener("click", scheme_right.bind(this));
265.        function scheme_right() {
266.            if (scheme_type == 2) {
267.                scheme_type = 1;
268.            }
269.            else {
270.                scheme_type += 1;
271.            }
272.        }
273.
274.
275.        this.SLeft2.addEventListener("click", meth_left.bind(this));
276.        function meth_left() {
277.            if (meth_type == 1) {
278.                meth_type = 3;
279.            }
280.            else {
281.                meth_type -= 1;
282.            }
283.        }
284.
285.        this.SRight2.addEventListener("click", meth_right.bind(this));
286.        function meth_right() {
287.            if (meth_type == 3) {
288.                meth_type = 1;
289.            }
290.            else {
291.                meth_type += 1;
292.            }
293.        }
294.
295.
296.        this.SLeft6.addEventListener("click", approx_left.bind(this));
297.        function approx_left() {
298.            if (approx_type == 1) {
```

```
                approx_type = 2;
            }
            else {
                approx_type -= 1;
            }
        }

        this.SRight6.addEventListener("click", approx_right.bind(this));
        function approx_right() {
            if (approx_type == 2) {
                approx_type = 1;
            }
            else {
                approx_type += 1;
            }
        }


        this.SLeft3.addEventListener("click", curStep_left.bind(this));
        function curStep_left() {
            if (s_cur > 0) {
                s_cur -= 1;
            }
        }

        this.SRight3.addEventListener("click", curStep_right.bind(this));
        function curStep_right() {
            if (s_cur < K) {
                s_cur += 1;
            }
        }


        this.stopBtn.addEventListener("click", curStep_stop.bind(this));
        function curStep_stop() {
            dt = 1000000;
        }

        this.SLeft4.addEventListener("click", pogr_left.bind(this));
        function pogr_left() {
            if (pogr_type == 1) {
                pogr_type = 3;
            }
            else {
                pogr_type -= 1;
            }
        }

        this.SRight4.addEventListener("click", pogr_right.bind(this));
        function pogr_right() {
            if (pogr_type == 3) {
                pogr_type = 1;
            }
            else {
                pogr_type += 1;
            }
        }

        this.SLeft5.addEventListener("click", scale_left.bind(this));
        function scale_left() {
            if (scale_type == 1) {
                scale_type = 2;
            }
            else {
                scale_type -= 1;
            }
        }

```

```
367.        this.SRight5.addEventListener("click", scale_right.bind(this));
368.        function scale_right() {
369.            if (scale_type == 2) {
370.                scale_type = 1;
371.            }
372.            else {
373.                scale_type += 1;
374.            }
375.        }
376.
377.        this.addEventListener("tick", setTexts2.bind(this));
378.        function setTexts2() {
379.            if (scheme_type == 1) {
380.                this.scheme_text.text = "Явная крест схема";
381.                theta = 1;
382.            }
383.            else if (scheme_type == 2) {
384.                this.scheme_text.text = "Неявная схема";
385.                theta = 0;
386.            }
387.
388.            if (meth_type == 1) {
389.                this.meth_text.text = "Двухточечная аппроксимация с первым порядком";
390.            }
391.            else if (meth_type == 2) {
392.                this.meth_text.text = "Трехточечная аппроксимация со вторым порядком";
393.            }
394.            else if (meth_type == 3) {
395.                this.meth_text.text = "Двухточечная аппроксимация со вторым порядком";
396.            }
397.
398.            if (approx_type == 1) {
399.                this.approx_text.text = "Первый порядок";
400.                theta = 1;
401.            }
402.            else if (approx_type == 2) {
403.                this.approx_text.text = "Второй порядок";
404.                theta = 0;
405.            }
406.
407.
408.            if (pogr_type == 1) {
409.                this.pogr_text.text = "Абсолютная погрешность";
410.            }
411.            else if (pogr_type == 2) {
412.                this.pogr_text.text = "Относительная погрешность";
413.            }
414.            else if (pogr_type == 3) {
415.                this.pogr_text.text = "Макс. Погрешность от времени";
416.            }
417.
418.
419.            if (scale_type == 1) {
420.                this.scale_text.text = "Макс. Фиксированный";
421.            }
422.            else if (scale_type == 2) {
423.                this.scale_text.text = "Динамический";
424.            }
425.
426.
427.            this.divDown0.text = "" + 0;
428.            this.divDown1.text = "" + Math.round(l*1/6*1000)/1000;
429.            this.divDown2.text = "" + Math.round(l*2/6*1000)/1000;
430.            this.divDown3.text = "" + Math.round(l*3/6*1000)/1000;
431.            this.divDown4.text = "" + Math.round(l*4/6*1000)/1000;
432.            this.divDown5.text = "" + Math.round(l*5/6*1000)/1000;
433.            this.divDown6.text = "" + Math.round(l*1000)/1000;
434.
```

```
435.
436.            if (pogr_type <= 2) {
437.                this.divDown0err.text = "" + 0;
438.                this.divDown1err.text = "" + Math.round(l*1/6*1000)/1000;
439.                this.divDown2err.text = "" + Math.round(l*2/6*1000)/1000;
440.                this.divDown3err.text = "" + Math.round(l*3/6*1000)/1000;
441.                this.divDown4err.text = "" + Math.round(l*4/6*1000)/1000;
442.                this.divDown5err.text = "" + Math.round(l*5/6*1000)/1000;
443.                this.divDown6err.text = "" + Math.round(l*1000)/1000;
444.            }
445.            else {
446.                this.divDown0err.text = "" + 0;
447.                this.divDown1err.text = "" + Math.round(T*1/6*1000)/1000;
448.                this.divDown2err.text = "" + Math.round(T*2/6*1000)/1000;
449.                this.divDown3err.text = "" + Math.round(T*3/6*1000)/1000;
450.                this.divDown4err.text = "" + Math.round(T*4/6*1000)/1000;
451.                this.divDown5err.text = "" + Math.round(T*5/6*1000)/1000;
452.                this.divDown6err.text = "" + Math.round(T*1000)/1000;
453.            }
454.
455.
456.            if (maxVals.length > 0 && scale_type == 2) {
457.                this.divUp0.text = "" + Math.round(minVals[s_cur]*1000)/1000;
458.                this.divUp1.text = "" + Math.round((minVals[s_cur] + (maxVals[s_cur] -
    minVals[s_cur])*1/7)*1000)/1000;
459.                this.divUp2.text = "" + Math.round((minVals[s_cur] + (maxVals[s_cur] -
    minVals[s_cur])*2/7)*1000)/1000;
460.                this.divUp3.text = "" + Math.round((minVals[s_cur] + (maxVals[s_cur] -
    minVals[s_cur])*3/7)*1000)/1000;
461.                this.divUp4.text = "" + Math.round((minVals[s_cur] + (maxVals[s_cur] -
    minVals[s_cur])*4/7)*1000)/1000;
462.                this.divUp5.text = "" + Math.round((minVals[s_cur] + (maxVals[s_cur] -
    minVals[s_cur])*5/7)*1000)/1000;
463.                this.divUp6.text = "" + Math.round((minVals[s_cur] + (maxVals[s_cur] -
    minVals[s_cur])*6/7)*1000)/1000;
464.                this.divUp7.text = "" + Math.round(maxVals[s_cur]*1000)/1000;
465.            }
466.            else {
467.                this.divUp0.text = "" + Math.round(minMinVal*1000)/1000;
468.                this.divUp1.text = "" + Math.round((minMinVal + (maxMaxVal -
    minMinVal)*1/7)*1000)/1000;
469.                this.divUp2.text = "" + Math.round((minMinVal + (maxMaxVal -
    minMinVal)*2/7)*1000)/1000;
470.                this.divUp3.text = "" + Math.round((minMinVal + (maxMaxVal -
    minMinVal)*3/7)*1000)/1000;
471.                this.divUp4.text = "" + Math.round((minMinVal + (maxMaxVal -
    minMinVal)*4/7)*1000)/1000;
472.                this.divUp5.text = "" + Math.round((minMinVal + (maxMaxVal -
    minMinVal)*5/7)*1000)/1000;
473.                this.divUp6.text = "" + Math.round((minMinVal + (maxMaxVal -
    minMinVal)*6/7)*1000)/1000;
474.                this.divUp7.text = "" + Math.round(maxMaxVal*1000)/1000;
475.            }
476.
477.            if (maxValsE.length > 0 && scale_type == 2 && pogr_type <= 2) {
478.                this.divUp0err.text = "" + Math.round(minValsE[s_cur]*1000)/1000;
479.                this.divUp1err.text = "" + Math.round((minValsE[s_cur] + (maxValsE[s_cur] -
    minValsE[s_cur])*1/7)*1000)/1000;
480.                this.divUp2err.text = "" + Math.round((minValsE[s_cur] + (maxValsE[s_cur] -
    minValsE[s_cur])*2/7)*1000)/1000;
481.                this.divUp3err.text = "" + Math.round((minValsE[s_cur] + (maxValsE[s_cur] -
    minValsE[s_cur])*3/7)*1000)/1000;
482.                this.divUp4err.text = "" + Math.round((minValsE[s_cur] + (maxValsE[s_cur] -
    minValsE[s_cur])*4/7)*1000)/1000;
483.                this.divUp5err.text = "" + Math.round((minValsE[s_cur] + (maxValsE[s_cur] -
    minValsE[s_cur])*5/7)*1000)/1000;
484.                this.divUp6err.text = "" + Math.round((minValsE[s_cur] + (maxValsE[s_cur] -
    minValsE[s_cur])*6/7)*1000)/1000;
```

```
485.                this.divUp7err.text = "" + Math.round(maxValsE[s_cur]*1000)/1000;
486.            }
487.            else if (pogr_type <= 2) {
488.                this.divUp0err.text = "" + Math.round(minMinValE*1000)/1000;
489.                this.divUp1err.text = "" + Math.round((minMinValE + (maxMaxValE -
    minMinValE)*1/7)*1000)/1000;
490.                this.divUp2err.text = "" + Math.round((minMinValE + (maxMaxValE -
    minMinValE)*2/7)*1000)/1000;
491.                this.divUp3err.text = "" + Math.round((minMinValE + (maxMaxValE -
    minMinValE)*3/7)*1000)/1000;
492.                this.divUp4err.text = "" + Math.round((minMinValE + (maxMaxValE -
    minMinValE)*4/7)*1000)/1000;
493.                this.divUp5err.text = "" + Math.round((minMinValE + (maxMaxValE -
    minMinValE)*5/7)*1000)/1000;
494.                this.divUp6err.text = "" + Math.round((minMinValE + (maxMaxValE -
    minMinValE)*6/7)*1000)/1000;
495.                this.divUp7err.text = "" + Math.round(maxMaxValE*1000)/1000;
496.            }
497.            else {
498.                this.divUp0err.text = "" + Math.round(minEotT*10000)/10000;
499.                this.divUp1err.text = "" + Math.round((minEotT + (maxEotT -
    minEotT)*1/7)*10000)/10000;
500.                this.divUp2err.text = "" + Math.round((minEotT + (maxEotT -
    minEotT)*2/7)*10000)/10000;
501.                this.divUp3err.text = "" + Math.round((minEotT + (maxEotT -
    minEotT)*3/7)*10000)/10000;
502.                this.divUp4err.text = "" + Math.round((minEotT + (maxEotT -
    minEotT)*4/7)*10000)/10000;
503.                this.divUp5err.text = "" + Math.round((minEotT + (maxEotT -
    minEotT)*5/7)*10000)/10000;
504.                this.divUp6err.text = "" + Math.round((minEotT + (maxEotT -
    minEotT)*6/7)*10000)/10000;
505.                this.divUp7err.text = "" + Math.round(maxEotT*10000)/10000;
506.            }
507.
508.            t_cur = s_cur*t;
509.
510.            this.step_text.text = "t = " + Math.round(t_cur*10000)/10000 + " (шаг " +
    Math.round(s_cur) + ")";
511.
512.            this.axisDown.y = Math.min(Math.max(transfY(0), 26.4), 418.6);
513.
514.            this.axisDownErr.y = Math.min(Math.max(transfEY(0), 468), 700);
515.        }
516.
517.
518.        //baseGraph
519.        //x = 0 : posX = 60
520.        //y = 0 : posY = 418.6
521.
522.        //x = max : posX = 627
523.        //y = max : posY = 26.4
524.
525.        //errGraph
526.        //x = 0 : posX = 53.6
527.        //y = 0 : posY = 700
528.
529.        //x = max : posX = 397.8
530.        //y = max : posY = 468.4
531.
532.
533.
534.
535.
536.        var t_cur = 0;
537.        //текущий момент времени
538.
539.        var s_cur = 0;
```

```
540.        //текущий шаг
541.
542.
543.        var etalon = [];
544.        var xVector = [];
545.        var tVector = [];
546.
547.        var maxMaxVal = 1;
548.        var minMinVal = 0;
549.
550.        var maxMaxValE = 1; //link
551.        var minMinValE = 0; //link
552.        var maxMaxValE1 = 1;
553.        var minMinValE1 = 0;
554.        var maxMaxValE2 = 1;
555.        var minMinValE2 = 0;
556.
557.        var maxVals = [];
558.        var minVals = [];
559.
560.        var maxValsE = []; //link
561.        var minValsE = []; //link
562.        var maxValsE1 = [];
563.        var minValsE1 = [];
564.        var maxValsE2 = [];
565.        var minValsE2 = [];
566.
567.
568.
569.        var EotT = [];
570.
571.        this.addEventListener("tick", setError.bind(this));
572.        function setError() {
573.            if (pogr_type == 1) {
574.                maxMaxValE = maxMaxValE1;
575.                minMinValE = minMinValE1;
576.                maxValsE = maxValsE1;
577.                minValsE = minValsE1;
578.                errorGraph = absolute_error;
579.            }
580.            else {
581.                maxMaxValE = maxMaxValE2;
582.                minMinValE = minMinValE2;
583.                maxValsE = maxValsE2;
584.                minValsE = minValsE2;
585.                errorGraph = relative_error;
586.            }
587.        }
588.
589.        var solve = [];
590.
591.
592.        function copyV(vector) {
593.            var newV = [];
594.            for (var i = 0 ; i < vector.length ; ++i) {
595.                newV.push(vector[i]);
596.            }
597.            return newV;
598.        }
599.
600.
601.        var absolute_error = [];
602.        var relative_error = [];
603.        var errorGraph = [];
604.
605.        var minEotT = 0;
606.        var maxEotT = 1;
607.
```

```
608.
609.
610.        function explicit_step(i, vector0, vector1, vector2) {
611.            var curT = i * t;
612.
613.            //Сначала вычисляем серединку
614.            for (var j = 1 ; j < N ; ++j) {
615.                //данную формулу я долго выводил на бумаге, аппроксимируя
616.                //производные
617.
618.                vector2[j] = (-2*vector1[j]*(t*t - h*h - t*h*h - t*t*h
619.                    + 1.5*t*t*h*h) - vector0[j]*h*h + vector1[j+1]*t*t
620.                    - vector1[j-1]*t*t*(2*h-1)) / (h*h*(1 + 2*t));
621.
622.
623.
624.
625.                maxVals[i] = (vector1[j] > maxVals[i] ? vector2[j] : maxVals[i]);
626.                minVals[i] = (vector1[j] < minVals[i] ? vector2[j] : minVals[i]);
627.            }
628.
629.            //аппроксимируем граничные условия
630.            //Граничные условия 1 рода аппроксимируются с беск. порядком
631.            //Поэтому, выбор метода аппроксимации гр. усл. не меняет ничего
632.
633.
634.            vector2[0] = phi_0(curT)/beta;
635.            vector2[N] = 0;
636.
637.
638.            maxVals[i] = (vector2[0] > maxVals[i] ? vector2[0] : maxVals[i]);
639.            maxVals[i] = (vector2[N] > maxVals[i] ? vector2[N] : maxVals[i]);
640.
641.            minVals[i] = (vector2[0] < minVals[i] ? vector2[0] : minVals[i]);
642.            minVals[i] = (vector2[N] < minVals[i] ? vector2[N] : minVals[i]);
643.
644.            maxMaxVal = (maxVals[i] > maxMaxVal ? maxVals[i] : maxMaxVal);
645.            minMinVal = (minVals[i] < minMinVal ? minVals[i] : minMinVal);
646.
647.            return copyV(vector2);
648.
649.        }
650.
651.
652.
653.        function implicit_step(i, vector0, vector1, vector2) {
654.            var curT = i * t;
655.
656.            var solveMatrix = [];
657.            var solveVector = [];
658.
659.            var emptV = [];
660.            for (var j = 0 ; j <= N ; ++j) {
661.                emptV.push(0);
662.            }
663.
664.            var thisV = copyV(emptV);
665.
666.            thisV[0] = beta
667.            thisV[1] = 0;
668.
669.            solveMatrix.push(copyV(thisV));
670.
671.            var d0 = phi_0(curT);
672.
673.            solveVector.push(d0); //d0
674.
675.
```

```
676.            for (var j = 1 ; j < N ; ++j) {
677.                var thisV = copyV(emptV);
678.                thisV[j - 1] = -1/(h*h);
679.                thisV[j] = 1/(t*t) + 2/t + 2/(h*h) + 2/h + 3;
680.                thisV[j + 1] = -1/(h*h) - 2/h;
681.
682.                solveMatrix.push(copyV(thisV));
683.
684.                var dj = (2*vector1[j] - vector0[j])/(t*t) + 2*vector1[j]/t;
685.                solveVector.push(dj);
686.
687.
688.            }
689.
690.            var thisV = copyV(emptV);
691.
692.
693.            thisV[N-1] = 0;
694.            thisV[N] = delta;
695.
696.            solveMatrix.push(copyV(thisV));
697.
698.            var dn = 0;
699.
700.            solveVector.push(dn); //d1
701.
702.
703.            var the_vector = progonka(solveMatrix, solveVector);
704.
705.
706.            for (var j = 0 ; j <= N ; ++j) {
707.
708.                maxVals[i] = (the_vector[j] > maxVals[i] ? the_vector[j] : maxVals[i]);
709.                minVals[i] = (the_vector[j] < minVals[i] ? the_vector[j] : minVals[i]);
710.
711.                maxMaxVal = (maxVals[i] > maxMaxVal ? maxVals[i] : maxMaxVal);
712.                minMinVal = (minVals[i] < minMinVal ? minVals[i] : minMinVal);
713.
714.            }
715.
716.            return copyV(the_vector);
717.        }
718.
719.
720.
721.        this.beginBtn.addEventListener("click", beginSimulation.bind(this));
722.        function beginSimulation() {
723.            generation += 1;
724.            s_cur = 0;
725.            etalon = [];
726.            solve = [];
727.            maxMaxVal = 1;
728.            minMinVal = 0;
729.
730.            maxVals = [];
731.            minVals = [];
732.
733.            xVector = [];
734.            tVector = [];
735.
736.            for (var j = 0 ; j <= N ; ++j) {
737.                xVector.push(h * j);
738.            }
739.            for (var i = 0 ; i <= K ; ++i) {
740.                var timeVector = [];
741.                var this_t = t * i;
742.                tVector.push(this_t);
743.                var this_max = 0;
```

```
744.                var this_min = 0;
745.                for (var j = 0 ; j <= N ; ++j) {
746.                    var this_x = xVector[j];
747.                    var func_res = U(this_x, this_t);
748.                    if (j == 0) {
749.                        this_max = func_res;
750.                        this_min = func_res;
751.                    }
752.                    else {
753.                        this_max = (func_res > this_max ? func_res : this_max);
754.                        this_min = (func_res < this_min ? func_res : this_min);
755.                    }
756.                    timeVector.push(func_res);
757.                }
758.                etalon.push(timeVector);
759.                maxVals.push(this_max);
760.                minVals.push(this_min);
761.                if (i == 0) {
762.                    maxMaxVal = this_max;
763.                    minMinVal = this_min;
764.                }
765.                else {
766.                    maxMaxVal = (this_max > maxMaxVal ? this_max : maxMaxVal);
767.                    minMinVal = (this_min < minMinVal ? this_min : minMinVal);
768.                }
769.            }
770.
771.            //задаем начальное условие
772.            var psiGrid0 = [];
773.            var emptV = [];
774.            for (var j = 0 ; j <= N ; ++j) {
775.                psiGrid0.push(psi(xVector[j]));
776.                emptV.push(0);
777.            }
778.            solve.push(psiGrid0);
779.
780.            var psiGrid1 = [];
781.            //1 порядок аппрокс. 2 нач условия
782.            if (approx_type == 1) {
783.                for (var j = 0 ; j <= N ; ++j) {
784.                    var ans = psi(xVector[j]) + psi2(xVector[j])*t;
785.                    psiGrid1.push(ans);
786.                }
787.                solve.push(psiGrid1);
788.            }
789.            else if (approx_type == 2) {
790.                for (var j = 0 ; j <= N ; ++j) {
791.                    var ans = psi(xVector[j]) + psi2(xVector[j])*t + (a*a*psi2d2(xVector[j]) +
    b*psi2d1(xVector[j]) + c*psi2(xVector[j]))*(t*t/2);
792.                    psiGrid1.push(ans);
793.                }
794.                solve.push(psiGrid1);
795.            }
796.
797.
798.
799.
800.            var vector0 = psiGrid0;
801.            var vector1 = psiGrid1;
802.            var vector2 = copyV(emptV);
803.
804.            //scheme 1
805.            if (scheme_type == 1) {
806.
807.                for (var i = 2 ; i <= K ; ++i) {
808.
809.                    var resV = explicit_step(i, vector0, vector1, vector2);
810.
```

```
811.                    solve.push(copyV(resV));
812.
813.                    vector0 = copyV(vector1);
814.                    vector1 = copyV(resV);
815.                    vector2 = copyV(emptV);
816.
817.                }
818.
819.            }
820.            else if (scheme_type == 2) {
821.
822.                for (var i = 2 ; i <= K ; ++i) {
823.
824.                    var resV = implicit_step(i, vector0, vector1, vector2);
825.
826.                    solve.push(copyV(resV));
827.
828.                    vector0 = copyV(vector1);
829.                    vector1 = copyV(resV);
830.                    vector2 = copyV(emptV);
831.
832.                }
833.            }
834.
835.
836.
837.            // error
838.
839.
840.            absolute_error = [];
841.            relative_error = [];
842.            maxMaxValE1 = 1;
843.            minMinValE1 = 0;
844.            maxMaxValE2 = 1;
845.            minMinValE2 = 0;
846.
847.            maxValsE1 = [];
848.            minValsE1 = [];
849.            maxValsE2 = [];
850.            minValsE2 = [];
851.
852.            EotT = [];
853.
854.            for (var i = 0 ; i <= K ; ++i) {
855.                var timeVector1 = [];
856.                var timeVector2 = [];
857.                var this_t = t * i;
858.                var this_max1 = 0;
859.                var this_min1 = 0;
860.
861.                var this_max2 = 0;
862.                var this_min2 = 0;
863.
864.                for (var j = 0 ; j <= N ; ++j) {
865.                    var this_x = xVector[j];
866.
867.                    var err1 = solve[i][j] - etalon[i][j];
868.                    var err2 = 0;
869.                    if (Math.abs(etalon[i][j]) > 0.1) {
870.                        err2 = Math.abs(1 - solve[i][j]/etalon[i][j]);
871.                    }
872.                    else {
873.                        err2 = Math.abs(1 - (solve[i][j]+0.2)/(etalon[i][j] + 0.2));
874.                    }
875.
876.                    if (j == 0) {
877.                        this_max1 = err1;
878.                        this_min1 = err1;
```

```
879.
880.                     this_max2 = err2;
881.                     this_min2 = err2;
882.                 }
883.                 else {
884.                     this_max1 = (err1 > this_max1 ? err1 : this_max1);
885.                     this_min1 = (err1 < this_min1 ? err1 : this_min1);
886.
887.                     this_max2 = (err2 > this_max2 ? err2 : this_max2);
888.                     this_min2 = (err2 < this_min2 ? err2 : this_min2);
889.                 }
890.                 timeVector1.push(err1);
891.                 timeVector2.push(err2);
892.             }
893.
894.             absolute_error.push(timeVector1);
895.             relative_error.push(timeVector2);
896.
897.             maxValsE1.push(this_max1);
898.             minValsE1.push(this_min1);
899.
900.             maxValsE2.push(this_max2);
901.             minValsE2.push(this_min2);
902.
903.
904.             if (i == 0) {
905.                 maxMaxValE1 = this_max1;
906.                 minMinValE1 = this_min1;
907.
908.                 maxMaxValE2 = this_max2;
909.                 minMinValE2 = this_min2;
910.
911.                 var stepEot = Math.max(maxMaxValE1, Math.abs(minMinValE1));
912.
913.                 EotT.push(stepEot);
914.
915.                 minEotT = stepEot;
916.                 maxEotT = stepEot;
917.             }
918.             else {
919.                 maxMaxValE1 = (this_max1 > maxMaxValE1 ? this_max1 : maxMaxValE1);
920.                 minMinValE1 = (this_min1 < minMinValE1 ? this_min1 : minMinValE1);
921.
922.                 var stepEot = Math.max(this_max1, Math.abs(this_min1));
923.
924.                 minEotT = (stepEot < minEotT ? stepEot : minEotT);
925.                 maxEotT = (stepEot > maxEotT ? stepEot : maxEotT);
926.
927.                 EotT.push(stepEot);
928.
929.                 maxMaxValE2 = (this_max2 > maxMaxValE2 ? this_max2 : maxMaxValE2);
930.                 minMinValE2 = (this_min2 < minMinValE2 ? this_min2 : minMinValE2);
931.             }
932.         }
933.
934.
935.
936.         makeEtalonGraph();
937.         makeSolveGraph();
938.         makeErrorGraph();
939.         makeErrorTGraph();
940.
941.         go = true;
942.     }
943.
944.     function transfX(x) {
945.         var newX = 60 + (567 * (x / 1));
946.         return newX;
```

```
947.        }
948.
949.        function transfY(y) {
950.            var newY = 0;
951.            if (maxVals.length > 0 && scale_type == 2) {
952.                newY = 418.6 - (382.6 * (y - minVals[s_cur])/(maxVals[s_cur] - minVals[s_cur]));
953.            }
954.            else {
955.                newY = 418.6 - (382.6 * (y - minMinVal)/(maxMaxVal - minMinVal));
956.            }
957.            return newY;
958.        }
959.
960.
961.        function transfEX(x) {
962.            var newX = 60.6 + (344.2 * (x / l));
963.            return newX;
964.        }
965.
966.
967.        function transfE2X(x) {
968.            var newX = 60.6 + (344.2 * (x / T));
969.            return newX;
970.        }
971.
972.        function transfEY(y) {
973.            var newY = 0;
974.            if (maxValsE.length > 0 && scale_type == 2) {
975.                newY = 700 - (231 * (y - minValsE[s_cur])/(maxValsE[s_cur] - minValsE[s_cur]));
976.            }
977.            else {
978.                newY = 700 - (231 * (y - minMinValE)/(maxMaxValE - minMinValE));
979.            }
980.            return newY;
981.        }
982.
983.        function transfE2Y(y) {
984.            var newY = 0;
985.            newY = 700 - (231 * (y - minEotT)/(maxEotT - minEotT));
986.
987.            return newY;
988.        }
989.
990.
991.        function makeEtalonGraph() {
992.            for (var i = 0 ; i < N ; ++i) {
993.                var join = new lib.line();
994.                stage.addChild(join);
995.                join.x = transfX(xVector[i]);
996.                join.y = transfY(etalon[0][i]);
997.                join.endX = transfX(xVector[i+1]);
998.                join.endY = transfY(etalon[0][i+1]);
999.
1000.               join.gotoAndStop(0);
1001.               join.num = i;
1002.
1003.               join.len = Math.sqrt(Math.pow((join.endY - join.y), 2) + Math.pow((join.endX -
    join.x), 2));
1004.
1005.               join.scaleX = join.len;
1006.               join.scaleY = 1;
1007.
1008.               join.rotation = Math.atan2((join.endY - join.y), (join.endX - join.x)) * 180 /
    Math.PI;
1009.
1010.               join.gen = generation;
1011.
1012.
```

```
1013.               join.visible = true;
1014.               join.alpha = 1;
1015.
1016.               join.addEventListener('tick', setPoses);
1017.           }
1018.
1019.       }
1020.
1021.       function makeSolveGraph() {
1022.           for (var i = 0 ; i < N ; ++i) {
1023.               var join = new lib.line();
1024.               stage.addChild(join);
1025.               join.x = transfX(xVector[i]);
1026.               join.y = transfY(solve[0][i]);
1027.               join.endX = transfX(xVector[i+1]);
1028.               join.endY = transfY(solve[0][i+1]);
1029.
1030.               join.gotoAndStop(1);
1031.               join.num = i;
1032.
1033.               join.len = Math.sqrt(Math.pow((join.endY - join.y), 2) + Math.pow((join.endX -
    join.x), 2));
1034.
1035.               join.scaleX = join.len;
1036.               join.scaleY = 1;
1037.
1038.               join.rotation = Math.atan2((join.endY - join.y), (join.endX - join.x)) * 180 /
    Math.PI;
1039.
1040.               join.gen = generation;
1041.
1042.
1043.               join.visible = true;
1044.               join.alpha = 1;
1045.
1046.               join.addEventListener('tick', setPoses2);
1047.           }
1048.
1049.       }
1050.
1051.       function makeErrorGraph() {
1052.           setError();
1053.           for (var i = 0 ; i < N ; ++i) {
1054.               var join = new lib.line();
1055.               stage.addChild(join);
1056.               join.x = transfEX(xVector[i]);
1057.               join.y = transfEY(errorGraph[0][i]);
1058.               join.endX = transfEX(xVector[i+1]);
1059.               join.endY = transfEY(errorGraph[0][i+1]);
1060.
1061.               join.gotoAndStop(3);
1062.               join.num = i;
1063.
1064.               join.len = Math.sqrt(Math.pow((join.endY - join.y), 2) + Math.pow((join.endX -
    join.x), 2));
1065.
1066.               join.scaleX = join.len;
1067.               join.scaleY = 1;
1068.
1069.               join.rotation = Math.atan2((join.endY - join.y), (join.endX - join.x)) * 180 /
    Math.PI;
1070.
1071.               join.gen = generation;
1072.
1073.
1074.               join.visible = true;
1075.               join.alpha = 1;
1076.
```

```
1077.               join.addEventListener('tick', setPoses3);
1078.           }
1079.
1080.       }
1081.
1082.
1083.       function makeErrorTGraph() {
1084.           for (var i = 0 ; i < K ; ++i) {
1085.               var join = new lib.line();
1086.               stage.addChild(join);
1087.               join.x = transfE2X(tVector[i]);
1088.               join.y = transfE2Y(EotT[i]);
1089.               join.endX = transfE2X(tVector[i+1]);
1090.               join.endY = transfE2Y(tVector[i+1]);
1091.
1092.               join.gotoAndStop(3);
1093.               join.num = i;
1094.
1095.               join.len = Math.sqrt(Math.pow((join.endY - join.y), 2) + Math.pow((join.endX -
    join.x), 2));
1096.
1097.               join.scaleX = join.len;
1098.               join.scaleY = 1;
1099.
1100.               join.rotation = Math.atan2((join.endY - join.y), (join.endX - join.x)) * 180 /
    Math.PI;
1101.
1102.               join.gen = generation;
1103.
1104.
1105.               join.visible = true;
1106.               join.alpha = 1;
1107.
1108.               join.addEventListener('tick', setPoses4);
1109.           }
1110.
1111.       }
1112.
1113.       function setPoses(e) {
1114.           var object = e.currentTarget;
1115.           if (object.gen == generation) {
1116.               object.x = transfX(xVector[object.num]);
1117.               object.y = transfY(etalon[s_cur][object.num]);
1118.               object.endX = transfX(xVector[object.num+1]);
1119.               object.endY = transfY(etalon[s_cur][object.num+1]);
1120.
1121.               object.len = Math.sqrt(Math.pow((object.endY - object.y), 2) + Math.pow((object.endX -
    object.x), 2));
1122.
1123.               object.scaleX = object.len;
1124.               object.scaleY = 1;
1125.
1126.               object.rotation = Math.atan2((object.endY - object.y), (object.endX - object.x)) * 180
    / Math.PI;
1127.           }
1128.           else if (object.gen != generation) {
1129.               object.alpha -= 3/30;
1130.           }
1131.
1132.           if (object.alpha <= 0) {
1133.               object.alpha = 0;
1134.               object.visible = false;
1135.               object.removeEventListener('tick', setPoses);
1136.               stage.removeChild(object);
1137.           }
1138.
1139.       }
1140.
```

```
1141.
1142.      function setPoses2(e) {
1143.          var object = e.currentTarget;
1144.          if (object.gen == generation) {
1145.              object.x = transfX(xVector[object.num]);
1146.              object.y = transfY(solve[s_cur][object.num]);
1147.              object.endX = transfX(xVector[object.num+1]);
1148.              object.endY = transfY(solve[s_cur][object.num+1]);
1149.
1150.              object.len = Math.sqrt(Math.pow((object.endY - object.y), 2) + Math.pow((object.endX -
    object.x), 2));
1151.
1152.              object.scaleX = object.len;
1153.              object.scaleY = 1;
1154.
1155.              object.rotation = Math.atan2((object.endY - object.y), (object.endX - object.x)) * 180
    / Math.PI;
1156.          }
1157.          else if (object.gen != generation) {
1158.              object.alpha -= 3/30;
1159.          }
1160.
1161.          if (object.alpha <= 0) {
1162.              object.alpha = 0;
1163.              object.visible = false;
1164.              object.removeEventListener('tick', setPoses2);
1165.              stage.removeChild(object);
1166.          }
1167.
1168.      }
1169.
1170.
1171.      function setPoses3(e) {
1172.          var object = e.currentTarget;
1173.          if (object.gen == generation) {
1174.              object.x = transfEX(xVector[object.num]);
1175.              object.y = transfEY(errorGraph[s_cur][object.num]);
1176.              object.endX = transfEX(xVector[object.num+1]);
1177.              object.endY = transfEY(errorGraph[s_cur][object.num+1]);
1178.
1179.              object.len = Math.sqrt(Math.pow((object.endY - object.y), 2) + Math.pow((object.endX -
    object.x), 2));
1180.
1181.              object.scaleX = object.len;
1182.              object.scaleY = 1;
1183.
1184.              object.rotation = Math.atan2((object.endY - object.y), (object.endX - object.x)) * 180
    / Math.PI;
1185.          }
1186.          else if (object.gen != generation) {
1187.              object.alpha -= 3/30;
1188.          }
1189.
1190.          if (pogr_type <= 2) {
1191.              object.visible = true;
1192.          }
1193.          else {
1194.              object.visible = false;
1195.          }
1196.
1197.          if (object.alpha <= 0) {
1198.              object.alpha = 0;
1199.              object.visible = false;
1200.              object.removeEventListener('tick', setPoses3);
1201.              stage.removeChild(object);
1202.          }
1203.
1204.      }
```

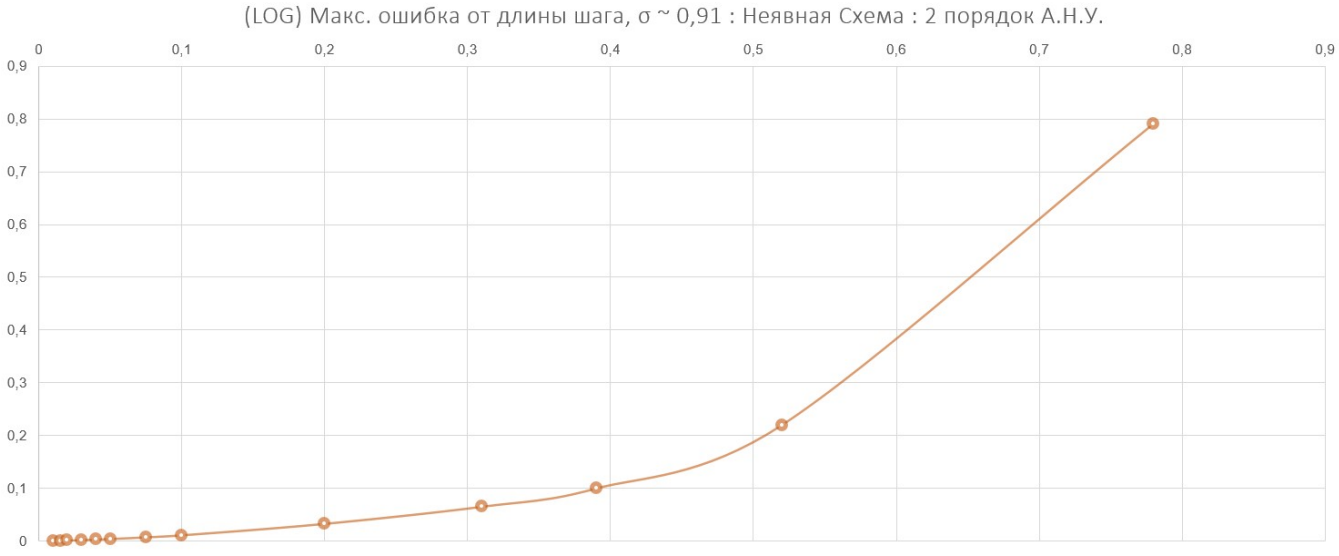
```
1205.
1206.
1207.       function setPoses4(e) {
1208.           var object = e.currentTarget;
1209.           if (object.gen == generation) {
1210.               object.x = transfE2X(tVector[object.num]);
1211.               object.y = transfE2Y(EotT[object.num]);
1212.               object.endX = transfE2X(tVector[object.num+1]);
1213.               object.endY = transfE2Y(EotT[object.num+1]);
1214.
1215.               object.len = Math.sqrt(Math.pow((object.endY - object.y), 2) + Math.pow((object.endX -
    object.x), 2));
1216.
1217.               object.scaleX = object.len;
1218.               object.scaleY = 1;
1219.
1220.               object.rotation = Math.atan2((object.endY - object.y), (object.endX - object.x)) * 180
    / Math.PI;
1221.           }
1222.           else if (object.gen != generation) {
1223.               object.alpha -= 3/30;
1224.           }
1225.
1226.           if (pogr_type == 3) {
1227.               object.visible = true;
1228.           }
1229.           else {
1230.               object.visible = false;
1231.           }
1232.
1233.           if (object.alpha <= 0) {
1234.               object.alpha = 0;
1235.               object.visible = false;
1236.               object.removeEventListener('tick', setPoses4);
1237.               stage.removeChild(object);
1238.           }
1239.
1240.       }
1241.
1242.
1243.
1244.
1245.
1246.
1247.
1248.
1249.       //метод прогонки
1250.       function progonka(matrix, vectorB) {
1251.           var vectorX = [];
1252.
1253.           var N = vectorB.length;
1254.
1255.           var alphas = [];
1256.           var betas = [];
1257.
1258.
1259.           for (var i = 0 ; i < vectorB.length ; ++i) {
1260.               alphas.push(0);
1261.               betas.push(0);
1262.           }
1263.
1264.           //Прямой ход прогонки
1265.
1266.           for (var i = 0; i < N; ++i) {
1267.               var A0, C0, B0, F0;
1268.
1269.               if (i - 1 < 0) {
1270.                   A0 = 0;
```

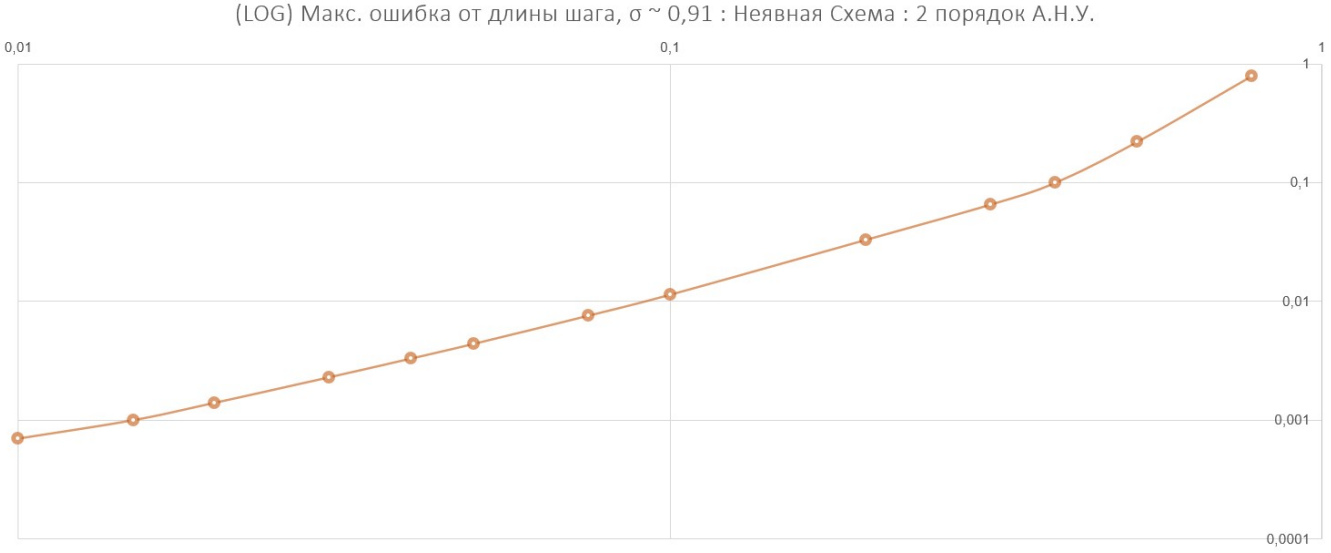
```
            }
            else {
                A0 = matrix[i][i - 1];
            }

            C0 = -1 * matrix[i][i];

            if (i + 1 < N) {
                B0 = matrix[i][i + 1];
            }
            else {
                B0 = 0;
            }

            F0 = vectorB[i];


            if (i == 0) {
                alphas[i] = B0 / C0;
                betas[i] = -(F0 / C0);
            }
            else if (i == N - 1) {
                alphas[i] = 0;
                betas[i] = (betas[i - 1] * A0 - F0) / (C0 - alphas[i - 1] * A0);
            }
            else {
                alphas[i] = B0 / (C0 - alphas[i - 1] * A0);
                betas[i] = (betas[i - 1] * A0 - F0) / (C0 - alphas[i - 1] * A0);
            }

        }


        vectorX[N - 1] = betas[N - 1];

        for (var i = 2; i <= N; ++i) {
            vectorX[N - i] = alphas[N - i] * vectorX[N - i + 1] + betas[N - i];
        }

        return vectorX;
    }
```

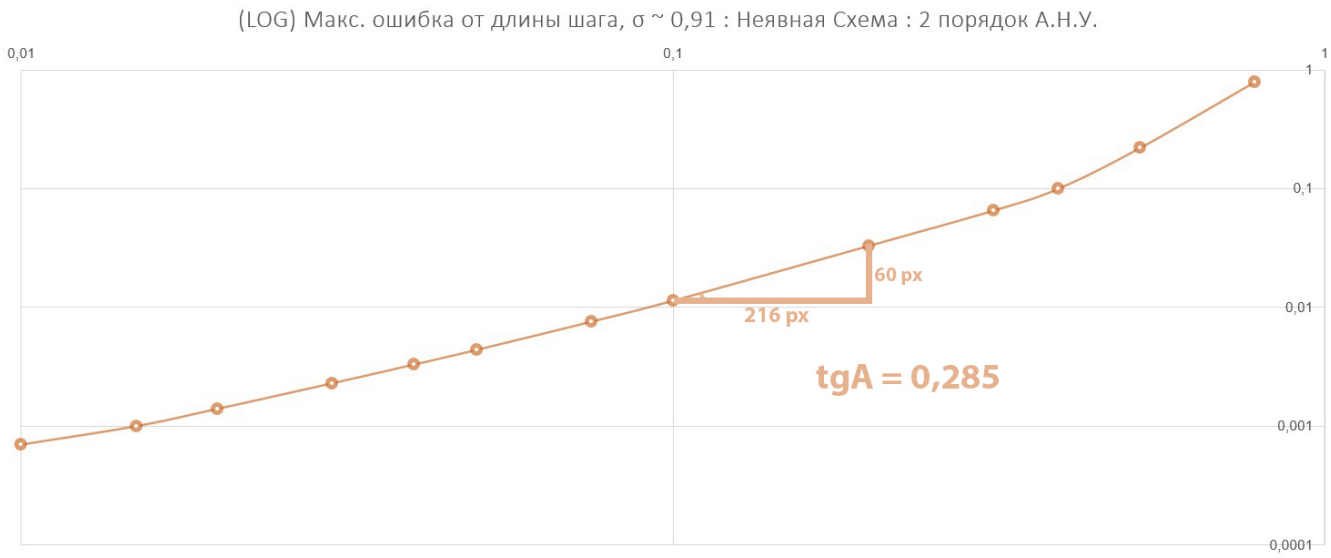
7. Графики

Для T = 4


График ошибки
0.0234
0.0201
0.0167
0.0134
0.01
0.0067
0.0033
0
0
0.667
1.333
2
2.667
3.333
4


(LOG) Макс. ошибка от длины шага, σ ~ 0,91 : Неявная Схема : 2 порядок А.Н.У.
0
0,1
0,2
0,3
0,4
0,5
0,6
0,7
0,8
0,9
0,9
0,8
0,7
0,6
0,5
0,4
0,3
0,2
0,1
0


(LOG) Макс. ошибка от длины шага, σ ~ 0,91 : Неявная Схема : 2 порядок А.Н.У.
0,01
0,1
1
1
0,1
0,01
0,001
0,0001



Данная лабораторная работа выполнена: **19 января 2021.**

Лабораторная работа 7

1. Тема ЛР:

   Метод конечных разностей для решения уравнений эллиптического типа. Уравнение Пуассона. Методы Либмана и Зейделя.

2. Вариант : **7**

7.

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = -2u,$$

$$u(0, y) = \cos y,$$

$$u(\frac{\pi}{2}, y) = 0,$$

$$u(x,0) = \cos x,$$

$$u(x, \frac{\pi}{2}) = 0.$$

Аналитическое решение: $U(x, y) = \cos x \cos y$.

3. Алгоритм:

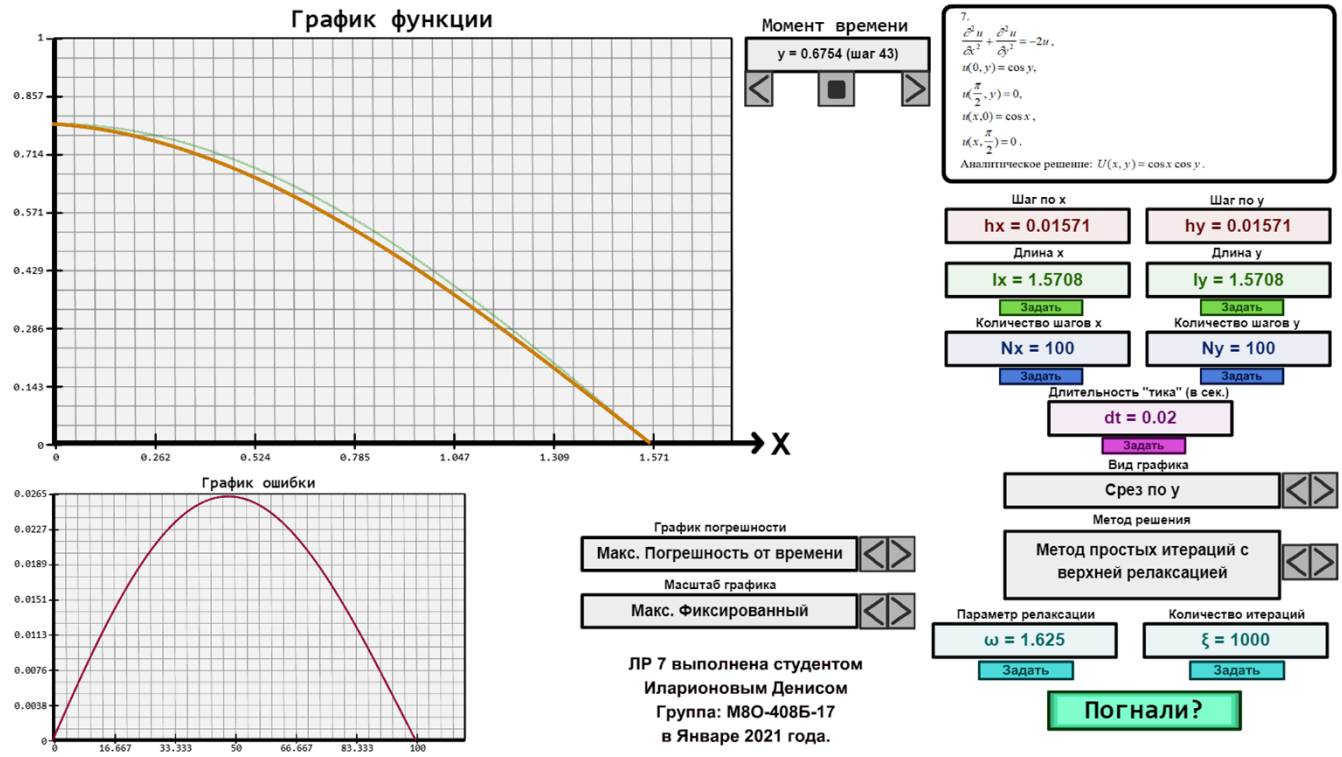
В данной лабораторной пришлось напрячься, потому что нужно отобразить срез как по X, так и по Y. То есть нужно выбрать одну из этих координат временной. Это заняло у меня много времени и сил. А сама лабораторная не очень сложная. Тут используются методы Либмана и Зейделя, которые были еще в 1 семестре. А также параметр релаксации. При $\omega < 1$ – метод Либмана. При $\omega = 1$ – метод Зейделя. А при $\omega > 1$ - метод Либмана с верхней релаксацией, который показывает наилучший результат, но параметр не должен быть близко к двойке или больше или равен ей, тогда алгоритм становится неустойчивым.

4. Среда разработки:

Adobe Animate, Javascript + HTML5

5. Реализация

Можно выбрать какую координату использовать временной + все, что в прошлых лабораторных. Границы аппроксимировать, к счастью, снова не надо (повезло с вариантом).



Вместо точности, я указал количество итераций. Думаю, так более логично и безопасно. Потому что при большом количестве шагов, чтобы добиться высокой точности необходимо огромное число итераций, что программа просто зависнет. Даже при максимальном параметре релаксации. Поэтому, количество итераций тут использовать удобнее, я считаю так.

6. Код программы

```
var alpha = 0; //α
var beta = 1; //β
var gamma = 0; //γ
var delta = 1; //δ

//Обозначим все как граничные условия 3 рода

function phi_x0(y) {
    return Math.cos(y);
}

function phi_xl(y) {
    return 0;
}

function phi_y0(x) {
    return Math.cos(x);
}

function phi_yl(x) {
    return 0;
}


function U(x, y) {
    return Math.cos(x) * Math.cos(y);
```

```
28. } //Аналитическое решение
29.
30.
31. //Параметры сетки
32. var lx = 3.1415926535/2; //интервал x
33. var Nx = 20; //количество шагов x
34. var Ny = 20; //количество шагов y
35. var ly = 3.1415926535/2; //интервал y
36.
37. var hx = 0; var hy = 0;
38.
39. var a = 1; var b = 2; var c = -3;
40.
41. var s1_cur = 0; //по y
42. var s2_cur = 0; //по x
43.
44. var dt = 0.02; //длительность тика
45. var cur_tick = 0;
46.
47.
48. var generation = 0; //номер генерации
49. var go = false; //увеличивать счетчик тиков?
50.
51.
52. var setupType = 1;
53. //1 срез по y
54. //2 срез по x
55.
56. var iters = 1000;
57. //Количество итераций
58.
59.
60. this.addEventListener("tick", det_ht.bind(this));
61. function det_ht() {
62.     hx = lx / Nx;
63.     hy = ly / Ny;
64.
65.     //определяем шаги по времени и пространству
66.
67.     this.h_text.text = "hx = " + Math.round(hx * 100000)/100000;
68.     this.tau_text.text = "hy = " + Math.round(hy * 100000)/100000;
69.
70.     this.l_text.text = "lx = " + Math.round(lx * 100000)/100000;
71.     this.T_text.text = "ly = " + Math.round(ly * 100000)/100000;
72.
73.     this.N_text.text = "Nx = " + Math.round(Nx);
74.     this.K_text.text = "Ny = " + Math.round(Ny);
75.
76.     this.dt_text.text = "dt = " + Math.round(dt*1000)/1000;
77.
78.     this.om_text.text = "ω = " + Math.round(relax*1000)/1000;
79.     this.it_text.text = "ξ = " + Math.round(iters);
80.
81.     if (setupType == 1) {
82.         if (go) {
83.             cur_tick += 1/60;
84.             if (cur_tick >= dt && s1_cur < Ny) {
85.                 var tGot = Math.floor(cur_tick / dt);
86.                 cur_tick %= dt;
87.                 s1_cur += Math.min(Ny - s1_cur, tGot);
88.             }
89.             else if (s1_cur == Ny) {
90.                 go = false;
91.             }
92.         }
93.     }
94.     else if (setupType == 2) {
95.         if (go) {
```

```
            cur_tick += 1/60;
            if (cur_tick >= dt && s2_cur < Nx) {
                var tGot = Math.floor(cur_tick / dt);
                cur_tick %= dt;
                        s2_cur += Math.min(Nx - s2_cur, tGot);
                    }
                    else if (s2_cur == Nx) {
                        go = false;
                    }
                }
            }

        }




        this.setGreen1.addEventListener("click", set_l_prompt.bind(this));
        function set_l_prompt() {
            generation += 1;
            var temp = prompt("Введите lx:", '');
            temp = Number.parseFloat(temp);
            if (isNaN(temp)) {
                //если ввели какую-то хрень
                temp = lx;
            }
            else if (temp > 1000000) {
                //ограничение на величину
                temp = 1000000;
            }
            else if (temp <= 0) {
                //всегда положительно
                temp = 0.001;
            }
            lx = temp;
        }



        this.setGreen2.addEventListener("click", set_T_prompt.bind(this));
        function set_T_prompt() {
            generation += 1;
            var temp = prompt("Введите ly:", '');
            temp = Number.parseFloat(temp);
            if (isNaN(temp)) {
                //если ввели какую-то хрень
                temp = ly;
            }
            else if (temp > 1000000) {
                //ограничение на величину
                temp = 1000000;
            }
            else if (temp <= 0) {
                //всегда положительно
                temp = 0.001;
            }
            ly = temp;
        }


        this.setBlue1.addEventListener("click", set_N_prompt.bind(this));
        function set_N_prompt() {
            generation += 1;
            var temp = prompt("Введите Nx:", '');
            temp = Number.parseInt(temp);
            if (isNaN(temp)) {
                //если ввели какую-то хрень
                temp = Nx;
```

```
            }
            else if (temp > 200) {
                //ограничение на величину
                temp = 200;
            }
            else if (temp <= 0) {
                //всегда положительно
                temp = 1;
            }
            Nx = temp;
        }


        this.setBlue2.addEventListener("click", set_K_prompt.bind(this));
        function set_K_prompt() {
            generation += 1;
            var temp = prompt("Введите Ny:", '');
            temp = Number.parseInt(temp);
            if (isNaN(temp)) {
                //если ввели какую-то хрень
                temp = Ny;
            }
            else if (temp > 200) {
                //ограничение на величину
                temp = 200;
            }
            else if (temp <= 0) {
                //всегда положительно
                temp = 1;
            }
            Ny = temp;
        }


        this.setPurp1.addEventListener("click", set_dt_prompt.bind(this));
        function set_dt_prompt() {
            cur_tick = 0;
            var temp = prompt("Введите dt (для ручного переключения можно ввести очень большим):",
    '');
            temp = Number.parseFloat(temp);
            if (isNaN(temp)) {
                //если ввели какую-то хрень
                temp = dt;
            }
            else if (temp > 1000000) {
                //ограничение на величину
                temp = 1000000;
            }
            else if (temp < 0.02) {
                //всегда положительно и больше 0.2 (почти плавная смена графика)
                temp = 0.02;
            }
            dt = temp;
        }


        this.setMag1.addEventListener("click", set_omega_prompt.bind(this));
        function set_omega_prompt() {
            generation += 1;
            var temp = prompt("Введите ω:", '');
            temp = Number.parseFloat(temp);
            if (isNaN(temp)) {
                //если ввели какую-то хрень
                temp = relax;
            }
            else if (temp > 1.99) {
                //ограничение на величину
                temp = 1.99;
```

```
231.            }
232.            else if (temp <= 0) {
233.                //всегда положительно
234.                temp = 0.001;
235.            }
236.            relax = temp;
237.        }
238.
239.
240.        this.setMag2.addEventListener("click", set_i_prompt.bind(this));
241.        function set_i_prompt() {
242.            generation += 1;
243.            var temp = prompt("Введите ξ:", '');
244.            temp = Number.parseInt(temp);
245.            if (isNaN(temp)) {
246.                //если ввели какую-то хрень
247.                temp = iters;
248.            }
249.            else if (temp > 10000) {
250.                //ограничение на величину
251.                temp = 10000;
252.            }
253.            else if (temp <= 0) {
254.                //всегда положительно
255.                temp = 1;
256.            }
257.            iters = temp;
258.        }
259.
260.        //Тип схемы - центрально-разностная
261.        //1,2 - Явная/Неявная схема
262.
263.        var relax = 1; //параметр релаксации
264.
265.
266.        var meth_type = 1;
267.        //Метод решения
268.        //1 - метод Либмана
269.        //2 - метод Зейделя
270.        //3 - метод Либмана с верхней релаксации
271.
272.        var approx_type = 1;
273.        //Метод аппроксимации 2 н.у
274.        //1 - Первый порядок
275.        //2 - Второй порядок
276.
277.
278.
279.        var pogr_type = 1;
280.        //Тип погрешности
281.        //1 - Абсолютная
282.        //2 - Относительная
283.        //3 - От времени
284.
285.        var scale_type = 1;
286.        //Масштаб графика
287.        //1 - Фиксированный
288.        //2 - Динамический
289.
290.        this.SLeft1.addEventListener("click", scheme_left.bind(this));
291.        function scheme_left() {
292.            if (setupType == 1) {
293.                setupType = 2;
294.            }
295.            else {
296.                setupType -= 1;
297.            }
298.        }
```

```
299.
300.        this.SRight1.addEventListener("click", scheme_right.bind(this));
301.        function scheme_right() {
302.            if (setupType == 2) {
303.                setupType = 1;
304.            }
305.            else {
306.                setupType += 1;
307.            }
308.        }
309.
310.
311.        this.SLeft2.addEventListener("click", meth_left.bind(this));
312.        function meth_left() {
313.            if (meth_type == 1) {
314.                meth_type = 3;
315.            }
316.            else {
317.                meth_type -= 1;
318.            }
319.        }
320.
321.        this.SRight2.addEventListener("click", meth_right.bind(this));
322.        function meth_right() {
323.            if (meth_type == 3) {
324.                meth_type = 1;
325.            }
326.            else {
327.                meth_type += 1;
328.            }
329.        }
330.
331.
332.
333.        this.SLeft3.addEventListener("click", curStep_left.bind(this));
334.        function curStep_left() {
335.            if (setupType == 1) {
336.                if (s1_cur > 0) {
337.                    s1_cur -= 1;
338.                }
339.            }
340.            else {
341.                if (s2_cur > 0) {
342.                    s2_cur -= 1;
343.                }
344.            }
345.        }
346.
347.        this.SRight3.addEventListener("click", curStep_right.bind(this));
348.        function curStep_right() {
349.            if (setupType == 1) {
350.                if (s1_cur < Ny) {
351.                    s1_cur += 1;
352.                }
353.            }
354.            else {
355.                if (s2_cur < Nx) {
356.                    s2_cur += 1;
357.                }
358.            }
359.        }
360.
361.
362.        this.stopBtn.addEventListener("click", curStep_stop.bind(this));
363.        function curStep_stop() {
364.            dt = 1000000;
365.        }
366.
```

```
367.        this.SLeft4.addEventListener("click", pogr_left.bind(this));
368.        function pogr_left() {
369.            if (pogr_type == 1) {
370.                pogr_type = 3;
371.            }
372.            else {
373.                pogr_type -= 1;
374.            }
375.        }
376.
377.        this.SRight4.addEventListener("click", pogr_right.bind(this));
378.        function pogr_right() {
379.            if (pogr_type == 3) {
380.                pogr_type = 1;
381.            }
382.            else {
383.                pogr_type += 1;
384.            }
385.        }
386.
387.        this.SLeft5.addEventListener("click", scale_left.bind(this));
388.        function scale_left() {
389.            if (scale_type == 1) {
390.                scale_type = 2;
391.            }
392.            else {
393.                scale_type -= 1;
394.            }
395.        }
396.
397.        this.SRight5.addEventListener("click", scale_right.bind(this));
398.        function scale_right() {
399.            if (scale_type == 2) {
400.                scale_type = 1;
401.            }
402.            else {
403.                scale_type += 1;
404.            }
405.        }
406.
407.        this.addEventListener("tick", setTexts2.bind(this));
408.        function setTexts2() {
409.            if (setupType == 1) {
410.                this.scheme_text.text = "Срез по y";
411.                theta = 1;
412.            }
413.            else if (setupType == 2) {
414.                this.scheme_text.text = "Срез по x";
415.                theta = 0;
416.            }
417.
418.            if (meth_type == 1) {
419.                this.meth_text.text = "Метод Либмана";
420.            }
421.            else if (meth_type == 2) {
422.                this.meth_text.text = "Метод Зейделя";
423.            }
424.            else if (meth_type == 3) {
425.                this.meth_text.text = "Метод простых итераций с верхней релаксацией";
426.            }
427.
428.
429.            if (pogr_type == 1) {
430.                this.pogr_text.text = "Абсолютная погрешность";
431.            }
432.            else if (pogr_type == 2) {
433.                this.pogr_text.text = "Относительная погрешность";
434.            }
```

```
435.            else if (pogr_type == 3) {
436.                this.pogr_text.text = "Макс. Погрешность от времени";
437.            }
438.
439.
440.            if (scale_type == 1) {
441.                this.scale_text.text = "Макс. Фиксированный";
442.            }
443.            else if (scale_type == 2) {
444.                this.scale_text.text = "Динамический";
445.            }
446.
447.
448.            if (setupType == 1) {
449.                this.divDown0.text = "" + 0;
450.                this.divDown1.text = "" + Math.round(lx*1/6*1000)/1000;
451.                this.divDown2.text = "" + Math.round(lx*2/6*1000)/1000;
452.                this.divDown3.text = "" + Math.round(lx*3/6*1000)/1000;
453.                this.divDown4.text = "" + Math.round(lx*4/6*1000)/1000;
454.                this.divDown5.text = "" + Math.round(lx*5/6*1000)/1000;
455.                this.divDown6.text = "" + Math.round(lx*1000)/1000;
456.            }
457.            else {
458.                this.divDown0.text = "" + 0;
459.                this.divDown1.text = "" + Math.round(ly*1/6*1000)/1000;
460.                this.divDown2.text = "" + Math.round(ly*2/6*1000)/1000;
461.                this.divDown3.text = "" + Math.round(ly*3/6*1000)/1000;
462.                this.divDown4.text = "" + Math.round(ly*4/6*1000)/1000;
463.                this.divDown5.text = "" + Math.round(ly*5/6*1000)/1000;
464.                this.divDown6.text = "" + Math.round(ly*1000)/1000;
465.            }
466.
467.            if (setupType == 1) {
468.                if (pogr_type <= 2) {
469.                    this.divDown0err.text = "" + 0;
470.                    this.divDown1err.text = "" + Math.round(lx*1/6*1000)/1000;
471.                    this.divDown2err.text = "" + Math.round(lx*2/6*1000)/1000;
472.                    this.divDown3err.text = "" + Math.round(lx*3/6*1000)/1000;
473.                    this.divDown4err.text = "" + Math.round(lx*4/6*1000)/1000;
474.                    this.divDown5err.text = "" + Math.round(lx*5/6*1000)/1000;
475.                    this.divDown6err.text = "" + Math.round(lx*1000)/1000;
476.                }
477.                else {
478.                    this.divDown0err.text = "" + 0;
479.                    this.divDown1err.text = "" + Math.round(Nx*1/6*1000)/1000;
480.                    this.divDown2err.text = "" + Math.round(Nx*2/6*1000)/1000;
481.                    this.divDown3err.text = "" + Math.round(Nx*3/6*1000)/1000;
482.                    this.divDown4err.text = "" + Math.round(Nx*4/6*1000)/1000;
483.                    this.divDown5err.text = "" + Math.round(Nx*5/6*1000)/1000;
484.                    this.divDown6err.text = "" + Math.round(Nx*1000)/1000;
485.                }
486.            }
487.            else {
488.                if (pogr_type <= 2) {
489.                    this.divDown0err.text = "" + 0;
490.                    this.divDown1err.text = "" + Math.round(ly*1/6*1000)/1000;
491.                    this.divDown2err.text = "" + Math.round(ly*2/6*1000)/1000;
492.                    this.divDown3err.text = "" + Math.round(ly*3/6*1000)/1000;
493.                    this.divDown4err.text = "" + Math.round(ly*4/6*1000)/1000;
494.                    this.divDown5err.text = "" + Math.round(ly*5/6*1000)/1000;
495.                    this.divDown6err.text = "" + Math.round(ly*1000)/1000;
496.                }
497.                else {
498.                    this.divDown0err.text = "" + 0;
499.                    this.divDown1err.text = "" + Math.round(Ny*1/6*1000)/1000;
500.                    this.divDown2err.text = "" + Math.round(Ny*2/6*1000)/1000;
501.                    this.divDown3err.text = "" + Math.round(Ny*3/6*1000)/1000;
502.                    this.divDown4err.text = "" + Math.round(Ny*4/6*1000)/1000;
```

```
503.                     this.divDown5err.text = "" + Math.round(Ny*5/6*1000)/1000;
504.                     this.divDown6err.text = "" + Math.round(Ny*1000)/1000;
505.                 }
506.             }
507.
508.             if (setupType == 1) {
509.                 if (maxValsx.length > 0 && scale_type == 2) {
510.                     this.divUp0.text = "" + Math.round(minValsx[s1_cur]*1000)/1000;
511.                     this.divUp1.text = "" + Math.round((minValsx[s1_cur] + (maxValsx[s1_cur] -
    minValsx[s1_cur])*1/7)*1000)/1000;
512.                     this.divUp2.text = "" + Math.round((minValsx[s1_cur] + (maxValsx[s1_cur] -
    minValsx[s1_cur])*2/7)*1000)/1000;
513.                     this.divUp3.text = "" + Math.round((minValsx[s1_cur] + (maxValsx[s1_cur] -
    minValsx[s1_cur])*3/7)*1000)/1000;
514.                     this.divUp4.text = "" + Math.round((minValsx[s1_cur] + (maxValsx[s1_cur] -
    minValsx[s1_cur])*4/7)*1000)/1000;
515.                     this.divUp5.text = "" + Math.round((minValsx[s1_cur] + (maxValsx[s1_cur] -
    minValsx[s1_cur])*5/7)*1000)/1000;
516.                     this.divUp6.text = "" + Math.round((minValsx[s1_cur] + (maxValsx[s1_cur] -
    minValsx[s1_cur])*6/7)*1000)/1000;
517.                     this.divUp7.text = "" + Math.round(maxValsx[s1_cur]*1000)/1000;
518.                 }
519.                 else {
520.                     this.divUp0.text = "" + Math.round(minMinValx*1000)/1000;
521.                     this.divUp1.text = "" + Math.round((minMinValx + (maxMaxValx -
    minMinValx)*1/7)*1000)/1000;
522.                     this.divUp2.text = "" + Math.round((minMinValx + (maxMaxValx -
    minMinValx)*2/7)*1000)/1000;
523.                     this.divUp3.text = "" + Math.round((minMinValx + (maxMaxValx -
    minMinValx)*3/7)*1000)/1000;
524.                     this.divUp4.text = "" + Math.round((minMinValx + (maxMaxValx -
    minMinValx)*4/7)*1000)/1000;
525.                     this.divUp5.text = "" + Math.round((minMinValx + (maxMaxValx -
    minMinValx)*5/7)*1000)/1000;
526.                     this.divUp6.text = "" + Math.round((minMinValx + (maxMaxValx -
    minMinValx)*6/7)*1000)/1000;
527.                     this.divUp7.text = "" + Math.round(maxMaxValx*1000)/1000;
528.                 }
529.             }
530.             else {
531.                 if (maxValsy.length > 0 && scale_type == 2) {
532.                     this.divUp0.text = "" + Math.round(minValsy[s2_cur]*1000)/1000;
533.                     this.divUp1.text = "" + Math.round((minValsy[s2_cur] + (maxValsy[s2_cur] -
    minValsy[s2_cur])*1/7)*1000)/1000;
534.                     this.divUp2.text = "" + Math.round((minValsy[s2_cur] + (maxValsy[s2_cur] -
    minValsy[s2_cur])*2/7)*1000)/1000;
535.                     this.divUp3.text = "" + Math.round((minValsy[s2_cur] + (maxValsy[s2_cur] -
    minValsy[s2_cur])*3/7)*1000)/1000;
536.                     this.divUp4.text = "" + Math.round((minValsy[s2_cur] + (maxValsy[s2_cur] -
    minValsy[s2_cur])*4/7)*1000)/1000;
537.                     this.divUp5.text = "" + Math.round((minValsy[s2_cur] + (maxValsy[s2_cur] -
    minValsy[s2_cur])*5/7)*1000)/1000;
538.                     this.divUp6.text = "" + Math.round((minValsy[s2_cur] + (maxValsy[s2_cur] -
    minValsy[s2_cur])*6/7)*1000)/1000;
539.                     this.divUp7.text = "" + Math.round(maxValsy[s2_cur]*1000)/1000;
540.                 }
541.                 else {
542.                     this.divUp0.text = "" + Math.round(minMinValy*1000)/1000;
543.                     this.divUp1.text = "" + Math.round((minMinValy + (maxMaxValy -
    minMinValy)*1/7)*1000)/1000;
544.                     this.divUp2.text = "" + Math.round((minMinValy + (maxMaxValy -
    minMinValy)*2/7)*1000)/1000;
545.                     this.divUp3.text = "" + Math.round((minMinValy + (maxMaxValy -
    minMinValy)*3/7)*1000)/1000;
546.                     this.divUp4.text = "" + Math.round((minMinValy + (maxMaxValy -
    minMinValy)*4/7)*1000)/1000;
547.                     this.divUp5.text = "" + Math.round((minMinValy + (maxMaxValy -
    minMinValy)*5/7)*1000)/1000;
```

```
548.                     this.divUp6.text = "" + Math.round((minMinValy + (maxMaxValy -
    minMinValy)*6/7)*1000)/1000;
549.                     this.divUp7.text = "" + Math.round(maxMaxValy*1000)/1000;
550.                 }
551.             }
552.
553.             if (setupType == 1) {
554.                 if (maxValsE.length > 0 && scale_type == 2 && pogr_type <= 2) {
555.                     this.divUp0err.text = "" + Math.round(minValsE[s1_cur]*1000)/1000;
556.                     this.divUp1err.text = "" + Math.round((minValsE[s1_cur] + (maxValsE[s1_cur] -
    minValsE[s1_cur])*1/7)*1000)/1000;
557.                     this.divUp2err.text = "" + Math.round((minValsE[s1_cur] + (maxValsE[s1_cur] -
    minValsE[s1_cur])*2/7)*1000)/1000;
558.                     this.divUp3err.text = "" + Math.round((minValsE[s1_cur] + (maxValsE[s1_cur] -
    minValsE[s1_cur])*3/7)*1000)/1000;
559.                     this.divUp4err.text = "" + Math.round((minValsE[s1_cur] + (maxValsE[s1_cur] -
    minValsE[s1_cur])*4/7)*1000)/1000;
560.                     this.divUp5err.text = "" + Math.round((minValsE[s1_cur] + (maxValsE[s1_cur] -
    minValsE[s1_cur])*5/7)*1000)/1000;
561.                     this.divUp6err.text = "" + Math.round((minValsE[s1_cur] + (maxValsE[s1_cur] -
    minValsE[s1_cur])*6/7)*1000)/1000;
562.                     this.divUp7err.text = "" + Math.round(maxValsE[s1_cur]*1000)/1000;
563.                 }
564.                 else if (pogr_type <= 2) {
565.                     this.divUp0err.text = "" + Math.round(minMinValE*1000)/1000;
566.                     this.divUp1err.text = "" + Math.round((minMinValE + (maxMaxValE -
    minMinValE)*1/7)*1000)/1000;
567.                     this.divUp2err.text = "" + Math.round((minMinValE + (maxMaxValE -
    minMinValE)*2/7)*1000)/1000;
568.                     this.divUp3err.text = "" + Math.round((minMinValE + (maxMaxValE -
    minMinValE)*3/7)*1000)/1000;
569.                     this.divUp4err.text = "" + Math.round((minMinValE + (maxMaxValE -
    minMinValE)*4/7)*1000)/1000;
570.                     this.divUp5err.text = "" + Math.round((minMinValE + (maxMaxValE -
    minMinValE)*5/7)*1000)/1000;
571.                     this.divUp6err.text = "" + Math.round((minMinValE + (maxMaxValE -
    minMinValE)*6/7)*1000)/1000;
572.                     this.divUp7err.text = "" + Math.round(maxMaxValE*1000)/1000;
573.                 }
574.                 else {
575.                     this.divUp0err.text = "" + Math.round(minEotTx*10000)/10000;
576.                     this.divUp1err.text = "" + Math.round((minEotTx + (maxEotTx -
    minEotTx)*1/7)*10000)/10000;
577.                     this.divUp2err.text = "" + Math.round((minEotTx + (maxEotTx -
    minEotTx)*2/7)*10000)/10000;
578.                     this.divUp3err.text = "" + Math.round((minEotTx + (maxEotTx -
    minEotTx)*3/7)*10000)/10000;
579.                     this.divUp4err.text = "" + Math.round((minEotTx + (maxEotTx -
    minEotTx)*4/7)*10000)/10000;
580.                     this.divUp5err.text = "" + Math.round((minEotTx + (maxEotTx -
    minEotTx)*5/7)*10000)/10000;
581.                     this.divUp6err.text = "" + Math.round((minEotTx + (maxEotTx -
    minEotTx)*6/7)*10000)/10000;
582.                     this.divUp7err.text = "" + Math.round(maxEotTx*10000)/10000;
583.                 }
584.             }
585.             else {
586.                 if (maxValsE.length > 0 && scale_type == 2 && pogr_type <= 2) {
587.                     this.divUp0err.text = "" + Math.round(minValsE[s2_cur]*1000)/1000;
588.                     this.divUp1err.text = "" + Math.round((minValsE[s2_cur] + (maxValsE[s2_cur] -
    minValsE[s2_cur])*1/7)*1000)/1000;
589.                     this.divUp2err.text = "" + Math.round((minValsE[s2_cur] + (maxValsE[s2_cur] -
    minValsE[s2_cur])*2/7)*1000)/1000;
590.                     this.divUp3err.text = "" + Math.round((minValsE[s2_cur] + (maxValsE[s2_cur] -
    minValsE[s2_cur])*3/7)*1000)/1000;
591.                     this.divUp4err.text = "" + Math.round((minValsE[s2_cur] + (maxValsE[s2_cur] -
    minValsE[s2_cur])*4/7)*1000)/1000;
```

```
592.                     this.divUp5err.text = "" + Math.round((minValsE[s2_cur] + (maxValsE[s2_cur] -
    minValsE[s2_cur])*5/7)*1000)/1000;
593.                     this.divUp6err.text = "" + Math.round((minValsE[s2_cur] + (maxValsE[s2_cur] -
    minValsE[s2_cur])*6/7)*1000)/1000;
594.                     this.divUp7err.text = "" + Math.round(maxValsE[s2_cur]*1000)/1000;
595.                 }
596.                 else if (pogr_type <= 2) {
597.                     this.divUp0err.text = "" + Math.round(minMinValE*1000)/1000;
598.                     this.divUp1err.text = "" + Math.round((minMinValE + (maxMaxValE -
    minMinValE)*1/7)*1000)/1000;
599.                     this.divUp2err.text = "" + Math.round((minMinValE + (maxMaxValE -
    minMinValE)*2/7)*1000)/1000;
600.                     this.divUp3err.text = "" + Math.round((minMinValE + (maxMaxValE -
    minMinValE)*3/7)*1000)/1000;
601.                     this.divUp4err.text = "" + Math.round((minMinValE + (maxMaxValE -
    minMinValE)*4/7)*1000)/1000;
602.                     this.divUp5err.text = "" + Math.round((minMinValE + (maxMaxValE -
    minMinValE)*5/7)*1000)/1000;
603.                     this.divUp6err.text = "" + Math.round((minMinValE + (maxMaxValE -
    minMinValE)*6/7)*1000)/1000;
604.                     this.divUp7err.text = "" + Math.round(maxMaxValE*1000)/1000;
605.                 }
606.                 else {
607.                     this.divUp0err.text = "" + Math.round(minEotTy*10000)/10000;
608.                     this.divUp1err.text = "" + Math.round((minEotTy + (maxEotTy -
    minEotTy)*1/7)*10000)/10000;
609.                     this.divUp2err.text = "" + Math.round((minEotTy + (maxEotTy -
    minEotTy)*2/7)*10000)/10000;
610.                     this.divUp3err.text = "" + Math.round((minEotTy + (maxEotTy -
    minEotTy)*3/7)*10000)/10000;
611.                     this.divUp4err.text = "" + Math.round((minEotTy + (maxEotTy -
    minEotTy)*4/7)*10000)/10000;
612.                     this.divUp5err.text = "" + Math.round((minEotTy + (maxEotTy -
    minEotTy)*5/7)*10000)/10000;
613.                     this.divUp6err.text = "" + Math.round((minEotTy + (maxEotTy -
    minEotTy)*6/7)*10000)/10000;
614.                     this.divUp7err.text = "" + Math.round(maxEotTy*10000)/10000;
615.                 }
616.             }
617.
618.
619.             if (setupType == 1) {
620.                 s1_cur2 = s1_cur*hy;
621.                 this.step_text.text = "y = " + Math.round(s1_cur2*10000)/10000 + " (шаг " +
    Math.round(s1_cur) + ")";
622.                 this.axisDown.axisTp.text = "X";
623.             }
624.             else {
625.                 s2_cur2 = s2_cur*hx;
626.                 this.step_text.text = "x = " + Math.round(s2_cur2*10000)/10000 + " (шаг " +
    Math.round(s2_cur) + ")";
627.                 this.axisDown.axisTp.text = "Y";
628.             }
629.             this.axisDown.y = Math.min(Math.max(transfY(0), 26.4), 418.6);
630.
631.
632.             this.axisDownErr.y = Math.min(Math.max(transfEY(0), 468), 700);
633.         }
634.
635.
636.         //baseGraph
637.         //x = 0 : posX = 60
638.         //y = 0 : posY = 418.6
639.
640.         //x = max : posX = 627
641.         //y = max : posY = 26.4
642.
643.         //errGraph
```

```
644.        //x = 0 : posX = 53.6
645.        //y = 0 : posY = 700
646.
647.        //x = max : posX = 397.8
648.        //y = max : posY = 468.4
649.
650.
651.
652.
653.
654.        var s1_cur = 0;
655.        //текущий шаг по y
656.
657.        var s2_cur = 0;
658.        //текущий шаг по x
659.
660.
661.        var etalon = [];
662.        var xVector = [];
663.        var yVector = [];
664.
665.        var maxMaxValx = 1;
666.        var minMinValx = 0;
667.
668.        var maxMaxValy = 1;
669.        var minMinValy = 0;
670.
671.        var maxMaxValE = 1; //link
672.        var minMinValE = 0; //link
673.
674.        var maxMaxValE1x = 1;
675.        var minMinValE1x = 0;
676.        var maxMaxValE2x = 1;
677.        var minMinValE2x = 0;
678.
679.        var maxMaxValE1y = 1;
680.        var minMinValE1y = 0;
681.        var maxMaxValE2y = 1;
682.        var minMinValE2y = 0;
683.
684.        var maxValsx = [];
685.        var minValsx = [];
686.
687.        var maxValsy = [];
688.        var minValsy = [];
689.
690.        var maxValsE = []; //link
691.        var minValsE = []; //link
692.
693.        var maxValsE1x = [];
694.        var minValsE1x = [];
695.        var maxValsE2x = [];
696.        var minValsE2x = [];
697.
698.        var maxValsE1y = [];
699.        var minValsE1y = [];
700.        var maxValsE2y = [];
701.        var minValsE2y = [];
702.
703.
704.
705.        var EotT = [];
706.
707.        this.addEventListener("tick", setError.bind(this));
708.        function setError() {
709.            if (setupType == 1) {
710.                if (pogr_type == 1) {
711.                    maxMaxValE = maxMaxValE1x;
```

```
                    minMinValE = minMinValE1x;
                    maxValsE = maxValsE1x;
                    minValsE = minValsE1x;
                    errorGraphX = absolute_errorX;
                    errorGraphY = absolute_errorY;
                }
                else {
                    maxMaxValE = maxMaxValE2x;
                    minMinValE = minMinValE2x;
                    maxValsE = maxValsE2x;
                    minValsE = minValsE2x;
                    errorGraphX = relative_errorX;
                    errorGraphY = relative_errorY;
                }
            }
            else {
                if (pogr_type == 1) {
                    maxMaxValE = maxMaxValE1y;
                    minMinValE = minMinValE1y;
                    maxValsE = maxValsE1y;
                    minValsE = minValsE1y;
                    errorGraphX = absolute_errorX;
                    errorGraphY = absolute_errorY;
                }
                else {
                    maxMaxValE = maxMaxValE2y;
                    minMinValE = minMinValE2y;
                    maxValsE = maxValsE2y;
                    minValsE = minValsE2y;
                    errorGraphX = relative_errorX;
                    errorGraphY = relative_errorY;
                }
            }
        }

        var solve = [];
        var solve2 = [];


        function copyV(vector) {
            var newV = [];
            for (var i = 0 ; i < vector.length ; ++i) {
                newV.push(vector[i]);
            }
            return newV;
        }

        function copyM(matrix) {
            var newM = [];
            for (var i = 0 ; i < matrix.length ; ++i) {
                newM.push(copyV(matrix[i]));
            }
            return newM;
        }

        var absolute_errorX = [];
        var relative_errorX = [];

        var absolute_errorY = [];
        var relative_errorY = [];
        var errorGraphX = [];
        var errorGraphY = [];

        var minEotTx = 0;
        var maxEotTx = 1;

        var minEotTy = 0;
        var maxEotTy = 1;
```

```
780.
781.
782.
783.
784.
785.        this.beginBtn.addEventListener("click", beginSimulation.bind(this));
786.        function beginSimulation() {
787.            generation += 1;
788.            s1_cur = 0;
789.            s2_cur = 0;
790.            etalon = [];
791.            solve = [];
792.            solve2 = [];
793.            maxMaxValx = 1;
794.            minMinValx = 0;
795.
796.            maxMaxValy = 1;
797.            minMinValy = 0;
798.
799.            maxValsx = [];
800.            minValsx = [];
801.
802.            maxValsy = [];
803.            minValsy = [];
804.
805.            xVector = [];
806.            yVector = [];
807.
808.            for (var j = 0 ; j <= Nx ; ++j) {
809.                xVector.push(hx * j);
810.            }
811.            for (var j = 0 ; j <= Ny ; ++j) {
812.                yVector.push(hy * j);
813.            }
814.            for (var i = 0 ; i <= Ny ; ++i) {
815.                var timeVector = [];
816.                var this_y = yVector[i];
817.                yVector.push(this_y);
818.                var this_maxX = 0;
819.                var this_minX = 0;
820.                for (var j = 0 ; j <= Nx ; ++j) {
821.                    var this_x = xVector[j];
822.                    var func_res = U(this_x, this_y);
823.                    if (j == 0) {
824.                        this_maxX = func_res;
825.                        this_minX = func_res;
826.                    }
827.                    else {
828.                        this_maxX = (func_res > this_maxX ? func_res : this_maxX);
829.                        this_minX = (func_res < this_minX ? func_res : this_minX);
830.                    }
831.                    if (i == 0) {
832.                        maxValsy.push(func_res);
833.                        minValsy.push(func_res);
834.                    }
835.                    else {
836.                        maxValsy[j] = (func_res > maxValsy[j] ? func_res : maxValsy[j]);
837.                        minValsy[j] = (func_res < minValsy[j] ? func_res : minValsy[j]);
838.                    }
839.
840.
841.                    timeVector.push(func_res);
842.                }
843.                etalon.push(timeVector);
844.                maxValsx.push(this_maxX);
845.                minValsx.push(this_minX);
846.                if (i == 0) {
847.                    maxMaxValx = this_maxX;
```

```
                    minMinValx = this_minX;
                }
                else {
                    maxMaxValx = (this_maxX > maxMaxValx ? this_maxX : maxMaxValx);
                    minMinValx = (this_minX < minMinValx ? this_minX : minMinValx);
                }
            }

            for (var i = 0 ; i < maxValsy.length ; ++i) {
                if (i == 0) {
                    maxMaxValy = maxValsy[i];
                    minMinValy = minValsy[i];
                }
                else {
                    maxMaxValy = (maxValsy[i] > maxMaxValy ? maxValsy[i] : maxMaxValy);
                    minMinValy = (minValsy[i] < minMinValy ? minValsy[i] : minMinValy);
                }
            }

            for (var p = 0 ; p <= Ny ; ++p) {
                var solvation = [];
                var curY = p*hy;
                for (var q = 0 ; q <= Nx ; ++q) {
                    var curX = q*hx;
                    if (p == 0) {
                        solvation.push(phi_y0(curX));
                    }
                    else if (p == Ny) {
                        solvation.push(phi_yl(curX));
                    }
                    else {
                        if (q == 0) {
                            solvation.push(phi_x0(curY));
                        }
                        else if (q == Nx) {
                            solvation.push(phi_xl(curY));
                        }
                        else {
                            solvation.push(0);
                        }
                    }
                }
                solve.push(copyV(solvation));
                solve2.push(copyV(solvation));
            }



            if (meth_type == 1) {
                for (var it = 0 ; it < iters ; ++it) {
                    for (var i = 1 ; i < Ny ; ++i) {
                        for (var j = 1 ; j < Nx ; ++j) {
                            solve2[i][j] = ((solve[i+1][j] + solve[i-1][j])*hy*hy
                            + (solve[i][j+1] + solve[i][j-1])*hx*hx) / (2*hx*hx + 2*hy*hy -
    2*hx*hx*hy*hy);
                        }
                    }
                    solve = copyM(solve2);
                }
            }


            if (meth_type == 2) {
                for (var it = 0 ; it < iters ; ++it) {
                    for (var i = 1 ; i < Ny ; ++i) {
                        for (var j = 1 ; j < Nx ; ++j) {
                            solve[i][j] = ((solve[i+1][j] + solve[i-1][j])*hy*hy
```

```
                            + (solve[i][j+1] + solve[i][j-1])*hx*hx) / (2*hx*hx + 2*hy*hy -
    2*hx*hx*hy*hy);
                        }
                    }
                }
            }

            if (meth_type == 3) {
                for (var it = 0 ; it < iters ; ++it) {
                    for (var i = 1 ; i < Ny ; ++i) {
                        for (var j = 1 ; j < Nx ; ++j) {
                            solve2[i][j] = ((solve[i+1][j] + solve[i-1][j])*hy*hy
                            + (solve[i][j+1] + solve[i][j-1])*hx*hx) / (2*hx*hx + 2*hy*hy -
    2*hx*hx*hy*hy);

                            solve[i][j] = relax*solve2[i][j] + (1-relax)*solve[i][j];
                        }
                    }
                }
            }

            for (var i = 0 ; i < maxValsx.length ; ++i) {
                for (var j = 0 ; j < maxValsy.length ; ++j) {
                    maxValsx[i] = (solve[i][j] > maxValsx[i] ? solve[i][j] : maxValsx[i]);
                    minValsx[i] = (solve[i][j] < minValsx[i] ? solve[i][j] : minValsx[i]);
                }
                maxMaxValx = (maxValsx[i] > maxMaxValx ? maxValsx[i] : maxMaxValx);
                minMinValx = (minValsx[i] < minMinValx ? minValsx[i] : minMinValx);
            }

            for (var i = 0 ; i < maxValsy.length ; ++i) {
                for (var j = 0 ; j < maxValsx.length ; ++j) {
                    maxValsy[i] = (solve[j][i] > maxValsy[i] ? solve[j][i] : maxValsy[i]);
                    minValsy[i] = (solve[j][i] < minValsy[i] ? solve[j][i] : minValsy[i]);
                }
                maxMaxValy = (maxValsy[i] > maxMaxValy ? maxValsy[i] : maxMaxValx);
                minMinValy = (minValsy[i] < minMinValy ? minValsy[i] : minMinValx);
            }



            // error


            absolute_errorX = [];
            relative_errorX = [];

            absolute_errorY = [];
            relative_errorY = [];

            maxMaxValE1x = 1;
            minMinValE1x = 0;
            maxMaxValE2x = 1;
            minMinValE2x = 0;

            maxMaxValE1y = 1;
            minMinValE1y = 0;
            maxMaxValE2y = 1;
            minMinValE2y = 0;

            maxValsE1x = [];
            minValsE1x = [];
            maxValsE2x = [];
            minValsE2x = [];

            maxValsE1y = [];
            minValsE1y = [];
            maxValsE2y = [];
```

```
980.            minValsE2y = [];
981.
982.            EotTx = [];
983.            EotTy = [];
984.
985.
986.            //error when y slice
987.            for (var i = 0 ; i <= Ny ; ++i) {
988.                var timeVector1 = [];
989.                var timeVector2 = [];
990.                var this_y = yVector[i]
991.                var this_max1 = 0;
992.                var this_min1 = 0;
993.
994.                var this_max2 = 0;
995.                var this_min2 = 0;
996.
997.                for (var j = 0 ; j <= Nx ; ++j) {
998.                    var this_x = xVector[j];
999.
1000.                   var err1 = solve[i][j] - etalon[i][j];
1001.                   var err2 = 0;
1002.                   if (Math.abs(etalon[i][j]) > 0.1) {
1003.                       err2 = Math.abs(1 - solve[i][j]/etalon[i][j]);
1004.                   }
1005.                   else {
1006.                       err2 = Math.abs(1 - (solve[i][j]+0.2)/(etalon[i][j] + 0.2));
1007.                   }
1008.
1009.                   if (j == 0) {
1010.                       this_max1 = err1;
1011.                       this_min1 = err1;
1012.
1013.                       this_max2 = err2;
1014.                       this_min2 = err2;
1015.                   }
1016.                   else {
1017.                       this_max1 = (err1 > this_max1 ? err1 : this_max1);
1018.                       this_min1 = (err1 < this_min1 ? err1 : this_min1);
1019.
1020.                       this_max2 = (err2 > this_max2 ? err2 : this_max2);
1021.                       this_min2 = (err2 < this_min2 ? err2 : this_min2);
1022.                   }
1023.                   timeVector1.push(err1);
1024.                   timeVector2.push(err2);
1025.               }
1026.
1027.               absolute_errorX.push(timeVector1);
1028.               relative_errorX.push(timeVector2);
1029.
1030.               maxValsE1x.push(this_max1);
1031.               minValsE1x.push(this_min1);
1032.
1033.               maxValsE2x.push(this_max2);
1034.               minValsE2x.push(this_min2);
1035.
1036.
1037.               if (i == 0) {
1038.                   maxMaxValE1x = this_max1;
1039.                   minMinValE1x = this_min1;
1040.
1041.                   maxMaxValE2x = this_max2;
1042.                   minMinValE2x = this_min2;
1043.
1044.                   var stepEotX = Math.max(maxMaxValE1x, Math.abs(minMinValE1x));
1045.
1046.                   EotTx.push(stepEotX);
1047.
```

```
                minEotTx = stepEotX;
                maxEotTx = stepEotX;
            }
            else {
                maxMaxValE1x = (this_max1 > maxMaxValE1x ? this_max1 : maxMaxValE1x);
                minMinValE1x = (this_min1 < minMinValE1x ? this_min1 : minMinValE1x);

                var stepEotX = Math.max(this_max1, Math.abs(this_min1));

                minEotTx = (stepEotX < minEotTx ? stepEotX : minEotTx);
                maxEotTx = (stepEotX > maxEotTx ? stepEotX : maxEotTx);

                EotTx.push(stepEotX);

                maxMaxValE2x = (this_max2 > maxMaxValE2x ? this_max2 : maxMaxValE2x);
                minMinValE2x = (this_min2 < minMinValE2x ? this_min2 : minMinValE2x);
            }
        }

        //error when x slice
        for (var i = 0 ; i <= Nx ; ++i) {
            var timeVector1 = [];
            var timeVector2 = [];
            var this_x = xVector[i]
            var this_max1 = 0;
            var this_min1 = 0;

            var this_max2 = 0;
            var this_min2 = 0;

            for (var j = 0 ; j <= Ny ; ++j) {
                var this_y = yVector[j];

                var err1 = solve[j][i] - etalon[j][i];
                var err2 = 0;
                if (Math.abs(etalon[j][i]) > 0.1) {
                    err2 = Math.abs(1 - solve[j][i]/etalon[j][i]);
                }
                else {
                    err2 = Math.abs(1 - (solve[j][i]+0.2)/(etalon[j][i] + 0.2));
                }

                if (j == 0) {
                    this_max1 = err1;
                    this_min1 = err1;

                    this_max2 = err2;
                    this_min2 = err2;
                }
                else {
                    this_max1 = (err1 > this_max1 ? err1 : this_max1);
                    this_min1 = (err1 < this_min1 ? err1 : this_min1);

                    this_max2 = (err2 > this_max2 ? err2 : this_max2);
                    this_min2 = (err2 < this_min2 ? err2 : this_min2);
                }
                timeVector1.push(err1);
                timeVector2.push(err2);
            }

            absolute_errorY.push(timeVector1);
            relative_errorY.push(timeVector2);

            maxValsE1y.push(this_max1);
            minValsE1y.push(this_min1);

            maxValsE2y.push(this_max2);
            minValsE2y.push(this_min2);
```

```
1116.
1117.
1118.             if (i == 0) {
1119.                 maxMaxValE1y = this_max1;
1120.                 minMinValE1y = this_min1;
1121.
1122.                 maxMaxValE2y = this_max2;
1123.                 minMinValE2y = this_min2;
1124.
1125.                 var stepEotY = Math.max(maxMaxValE1y, Math.abs(minMinValE1y));
1126.
1127.                 EotTy.push(stepEotY);
1128.
1129.                 minEotTy = stepEotY;
1130.                 maxEotTy = stepEotY;
1131.             }
1132.             else {
1133.                 maxMaxValE1y = (this_max1 > maxMaxValE1y ? this_max1 : maxMaxValE1y);
1134.                 minMinValE1y = (this_min1 < minMinValE1y ? this_min1 : minMinValE1y);
1135.
1136.                 var stepEotY = Math.max(this_max1, Math.abs(this_min1));
1137.
1138.                 minEotTy = (stepEotY < minEotTy ? stepEotY : minEotTy);
1139.                 maxEotTy = (stepEotY > maxEotTy ? stepEotY : maxEotTy);
1140.
1141.                 EotTy.push(stepEotY);
1142.
1143.                 maxMaxValE2y = (this_max2 > maxMaxValE2y ? this_max2 : maxMaxValE2y);
1144.                 minMinValE2y = (this_min2 < minMinValE2y ? this_min2 : minMinValE2y);
1145.             }
1146.         }
1147.
1148.         makeEtalonGraph();
1149.         makeSolveGraph();
1150.         makeErrorGraph();
1151.         makeErrorTGraph();
1152.
1153.         go = true;
1154.     }
1155.
1156.     function transfX(x) {
1157.         var newX;
1158.         if (setupType == 1) {
1159.             newX = 60 + (567 * (x / lx));
1160.         }
1161.         else {
1162.             newX = 60 + (567 * (x / ly));
1163.         }
1164.         return newX;
1165.     }
1166.
1167.     function transfY(y) {
1168.         var newY = 0;
1169.         if (setupType == 1) {
1170.             if (maxValsx.length > 0 && scale_type == 2) {
1171.                 newY = 418.6 - (382.6 * (y - minValsx[s1_cur])/(maxValsx[s1_cur] -
    minValsx[s1_cur]));
1172.             }
1173.             else {
1174.                 newY = 418.6 - (382.6 * (y - minMinValx)/(maxMaxValx - minMinValx));
1175.             }
1176.         }
1177.         else {
1178.             if (maxValsy.length > 0 && scale_type == 2) {
1179.                 newY = 418.6 - (382.6 * (y - minValsy[s2_cur])/(maxValsy[s2_cur] -
    minValsy[s2_cur]));
1180.             }
1181.             else {
```

```
1182.                 newY = 418.6 - (382.6 * (y - minMinValy)/(maxMaxValy - minMinValy));
1183.             }
1184.         }
1185.         return newY;
1186.     }
1187.
1188.
1189.     function transfEX(x) {
1190.         var newX;
1191.         if (setupType == 1) {
1192.             newX = 60.6 + (344.2 * (x / lx));
1193.         }
1194.         else {
1195.             newX = 60.6 + (344.2 * (x / ly));
1196.         }
1197.         return newX;
1198.     }
1199.
1200.
1201.     function transfE2X(x) {
1202.         var newX;
1203.         if (setupType == 1) {
1204.             newX = 60.6 + (344.2 * (x / ly));
1205.         }
1206.         else {
1207.             newX = 60.6 + (344.2 * (x / lx));
1208.         }
1209.         return newX;
1210.     }
1211.
1212.     function transfEY(y) {
1213.         var newY = 0;
1214.         if (setupType == 1) {
1215.             if (maxValsE.length > 0 && scale_type == 2) {
1216.                 newY = 700 - (231 * (y - minValsE[s1_cur])/(maxValsE[s1_cur] - minValsE[s1_cur]));
1217.             }
1218.             else {
1219.                 newY = 700 - (231 * (y - minMinValE)/(maxMaxValE - minMinValE));
1220.             }
1221.         }
1222.         else {
1223.             if (maxValsE.length > 0 && scale_type == 2) {
1224.                 newY = 700 - (231 * (y - minValsE[s2_cur])/(maxValsE[s2_cur] - minValsE[s2_cur]));
1225.             }
1226.             else {
1227.                 newY = 700 - (231 * (y - minMinValE)/(maxMaxValE - minMinValE));
1228.             }
1229.         }
1230.         return newY;
1231.     }
1232.
1233.     function transfE2Y(y) {
1234.         var newY = 0;
1235.         if (setupType == 1) {
1236.             newY = 700 - (231 * (y - minEotTx)/(maxEotTx - minEotTx));
1237.         }
1238.         else {
1239.             newY = 700 - (231 * (y - minEotTy)/(maxEotTy - minEotTy));
1240.         }
1241.
1242.         return newY;
1243.     }
1244.
1245.
1246.     function makeEtalonGraph() {
1247.         for (var i = 0 ; i < Nx ; ++i) {
1248.             var join = new lib.line();
1249.             stage.addChild(join);
```

```
1250.               join.x = transfX(xVector[i]);
1251.               join.y = transfY(etalon[0][i]);
1252.               join.endX = transfX(xVector[i+1]);
1253.               join.endY = transfY(etalon[0][i+1]);
1254.
1255.               join.gotoAndStop(0);
1256.               join.num = i;
1257.
1258.               join.len = Math.sqrt(Math.pow((join.endY - join.y), 2) + Math.pow((join.endX -
    join.x), 2));
1259.
1260.               join.scaleX = join.len;
1261.               join.scaleY = 1;
1262.
1263.               join.rotation = Math.atan2((join.endY - join.y), (join.endX - join.x)) * 180 /
    Math.PI;
1264.
1265.               join.gen = generation;
1266.
1267.
1268.               join.visible = true;
1269.               join.alpha = 1;
1270.
1271.               join.addEventListener('tick', setPoses11);
1272.           }
1273.           for (var i = 0 ; i < Ny ; ++i) {
1274.               var join = new lib.line();
1275.               stage.addChild(join);
1276.               join.x = transfX(yVector[i]);
1277.               join.y = transfY(etalon[i][0]);
1278.               join.endX = transfX(yVector[i+1]);
1279.               join.endY = transfY(etalon[i+1][0]);
1280.
1281.               join.gotoAndStop(0);
1282.               join.num = i;
1283.
1284.               join.len = Math.sqrt(Math.pow((join.endY - join.y), 2) + Math.pow((join.endX -
    join.x), 2));
1285.
1286.               join.scaleX = join.len;
1287.               join.scaleY = 1;
1288.
1289.               join.rotation = Math.atan2((join.endY - join.y), (join.endX - join.x)) * 180 /
    Math.PI;
1290.
1291.               join.gen = generation;
1292.
1293.
1294.               join.visible = true;
1295.               join.alpha = 1;
1296.
1297.               join.addEventListener('tick', setPoses12);
1298.           }
1299.
1300.       }
1301.
1302.       function makeSolveGraph() {
1303.           for (var i = 0 ; i < Nx ; ++i) {
1304.               var join = new lib.line();
1305.               stage.addChild(join);
1306.               join.x = transfX(xVector[i]);
1307.               join.y = transfY(solve[0][i]);
1308.               join.endX = transfX(xVector[i+1]);
1309.               join.endY = transfY(solve[0][i+1]);
1310.
1311.               join.gotoAndStop(1);
1312.               join.num = i;
1313.
```

```
1314.               join.len = Math.sqrt(Math.pow((join.endY - join.y), 2) + Math.pow((join.endX -
    join.x), 2));
1315.
1316.               join.scaleX = join.len;
1317.               join.scaleY = 1;
1318.
1319.               join.rotation = Math.atan2((join.endY - join.y), (join.endX - join.x)) * 180 /
    Math.PI;
1320.
1321.               join.gen = generation;
1322.
1323.
1324.               join.visible = true;
1325.               join.alpha = 1;
1326.
1327.               join.addEventListener('tick', setPoses21);
1328.           }
1329.           for (var i = 0 ; i < Ny ; ++i) {
1330.               var join = new lib.line();
1331.               stage.addChild(join);
1332.               join.x = transfX(yVector[i]);
1333.               join.y = transfY(solve[i][0]);
1334.               join.endX = transfX(yVector[i+1]);
1335.               join.endY = transfY(solve[i+1][0]);
1336.
1337.               join.gotoAndStop(1);
1338.               join.num = i;
1339.
1340.               join.len = Math.sqrt(Math.pow((join.endY - join.y), 2) + Math.pow((join.endX -
    join.x), 2));
1341.
1342.               join.scaleX = join.len;
1343.               join.scaleY = 1;
1344.
1345.               join.rotation = Math.atan2((join.endY - join.y), (join.endX - join.x)) * 180 /
    Math.PI;
1346.
1347.               join.gen = generation;
1348.
1349.
1350.               join.visible = true;
1351.               join.alpha = 1;
1352.
1353.               join.addEventListener('tick', setPoses22);
1354.           }
1355.       }
1356.
1357.       function makeErrorGraph() {
1358.           setError();
1359.           for (var i = 0 ; i < Nx ; ++i) {
1360.               var join = new lib.line();
1361.               stage.addChild(join);
1362.               join.x = transfEX(xVector[i]);
1363.               join.y = transfEY(errorGraphX[0][i]);
1364.               join.endX = transfEX(xVector[i+1]);
1365.               join.endY = transfEY(errorGraphX[0][i+1]);
1366.
1367.               join.gotoAndStop(3);
1368.               join.num = i;
1369.
1370.               join.len = Math.sqrt(Math.pow((join.endY - join.y), 2) + Math.pow((join.endX -
    join.x), 2));
1371.
1372.               join.scaleX = join.len;
1373.               join.scaleY = 1;
1374.
1375.               join.rotation = Math.atan2((join.endY - join.y), (join.endX - join.x)) * 180 /
    Math.PI;
```

```
1376.
1377.             join.gen = generation;
1378.
1379.
1380.             join.visible = true;
1381.             join.alpha = 1;
1382.
1383.             join.addEventListener('tick', setPoses31);
1384.         }
1385.
1386.         for (var i = 0 ; i < Ny ; ++i) {
1387.             var join = new lib.line();
1388.             stage.addChild(join);
1389.             join.x = transfEX(yVector[i]);
1390.             join.y = transfEY(errorGraphY[0][i]);
1391.             join.endX = transfEX(yVector[i+1]);
1392.             join.endY = transfEY(errorGraphY[0][i+1]);
1393.
1394.             join.gotoAndStop(3);
1395.             join.num = i;
1396.
1397.             join.len = Math.sqrt(Math.pow((join.endY - join.y), 2) + Math.pow((join.endX -
    join.x), 2));
1398.
1399.             join.scaleX = join.len;
1400.             join.scaleY = 1;
1401.
1402.             join.rotation = Math.atan2((join.endY - join.y), (join.endX - join.x)) * 180 /
    Math.PI;
1403.
1404.             join.gen = generation;
1405.
1406.
1407.             join.visible = true;
1408.             join.alpha = 1;
1409.
1410.             join.addEventListener('tick', setPoses32);
1411.         }
1412.
1413.
1414.     }
1415.
1416.
1417.     function makeErrorTGraph() {
1418.         for (var i = 0 ; i < Ny ; ++i) {
1419.             var join = new lib.line();
1420.             stage.addChild(join);
1421.             join.x = transfE2X(yVector[i]);
1422.             join.y = transfE2Y(EotTx[i]);
1423.             join.endX = transfE2X(yVector[i+1]);
1424.             join.endY = transfE2Y(EotTx[i+1]);
1425.
1426.             join.gotoAndStop(3);
1427.             join.num = i;
1428.
1429.             join.len = Math.sqrt(Math.pow((join.endY - join.y), 2) + Math.pow((join.endX -
    join.x), 2));
1430.
1431.             join.scaleX = join.len;
1432.             join.scaleY = 1;
1433.
1434.             join.rotation = Math.atan2((join.endY - join.y), (join.endX - join.x)) * 180 /
    Math.PI;
1435.
1436.             join.gen = generation;
1437.
1438.
1439.             join.visible = true;
```

```
1440.               join.alpha = 1;
1441.
1442.               join.addEventListener('tick', setPoses41);
1443.           }
1444.           for (var i = 0 ; i < Nx ; ++i) {
1445.               var join = new lib.line();
1446.               stage.addChild(join);
1447.               join.x = transfE2X(xVector[i]);
1448.               join.y = transfE2Y(EotTy[i]);
1449.               join.endX = transfE2X(xVector[i+1]);
1450.               join.endY = transfE2Y(EotTy[i+1]);
1451.
1452.               join.gotoAndStop(3);
1453.               join.num = i;
1454.
1455.               join.len = Math.sqrt(Math.pow((join.endY - join.y), 2) + Math.pow((join.endX -
    join.x), 2));
1456.
1457.               join.scaleX = join.len;
1458.               join.scaleY = 1;
1459.
1460.               join.rotation = Math.atan2((join.endY - join.y), (join.endX - join.x)) * 180 /
    Math.PI;
1461.
1462.               join.gen = generation;
1463.
1464.
1465.               join.visible = true;
1466.               join.alpha = 1;
1467.
1468.               join.addEventListener('tick', setPoses42);
1469.           }
1470.       }
1471.
1472.
1473.
1474.       function setPoses11(e) {
1475.           var object = e.currentTarget;
1476.           if (object.gen == generation) {
1477.               object.x = transfX(xVector[object.num]);
1478.               object.y = transfY(etalon[s1_cur][object.num]);
1479.               object.endX = transfX(xVector[object.num+1]);
1480.               object.endY = transfY(etalon[s1_cur][object.num+1]);
1481.
1482.               object.len = Math.sqrt(Math.pow((object.endY - object.y), 2) + Math.pow((object.endX -
    object.x), 2));
1483.
1484.               if (setupType == 1) {
1485.                   object.visible = true;
1486.               }
1487.               else {
1488.                   object.visible = false;
1489.               }
1490.
1491.
1492.               object.scaleX = object.len;
1493.               object.scaleY = 1;
1494.
1495.               object.rotation = Math.atan2((object.endY - object.y), (object.endX - object.x)) * 180
    / Math.PI;
1496.           }
1497.           else if (object.gen != generation) {
1498.               object.alpha -= 3/30;
1499.           }
1500.
1501.           if (object.alpha <= 0) {
1502.               object.alpha = 0;
1503.               object.visible = false;
```

```
1504.               object.removeEventListener('tick', setPoses11);
1505.               stage.removeChild(object);
1506.           }
1507.
1508.       }
1509.
1510.       function setPoses12(e) {
1511.           var object = e.currentTarget;
1512.           if (object.gen == generation) {
1513.               object.x = transfX(yVector[object.num]);
1514.               object.y = transfY(etalon[object.num][s2_cur]);
1515.               object.endX = transfX(yVector[object.num+1]);
1516.               object.endY = transfY(etalon[object.num+1][s2_cur]);
1517.
1518.               object.len = Math.sqrt(Math.pow((object.endY - object.y), 2) + Math.pow((object.endX -
    object.x), 2));
1519.
1520.               if (setupType == 2) {
1521.                   object.visible = true;
1522.               }
1523.               else {
1524.                   object.visible = false;
1525.               }
1526.
1527.
1528.               object.scaleX = object.len;
1529.               object.scaleY = 1;
1530.
1531.               object.rotation = Math.atan2((object.endY - object.y), (object.endX - object.x)) * 180
    / Math.PI;
1532.           }
1533.           else if (object.gen != generation) {
1534.               object.alpha -= 3/30;
1535.           }
1536.
1537.           if (object.alpha <= 0) {
1538.               object.alpha = 0;
1539.               object.visible = false;
1540.               object.removeEventListener('tick', setPoses12);
1541.               stage.removeChild(object);
1542.           }
1543.
1544.       }
1545.
1546.       function setPoses21(e) {
1547.           var object = e.currentTarget;
1548.           if (object.gen == generation) {
1549.               object.x = transfX(xVector[object.num]);
1550.               object.y = transfY(solve[s1_cur][object.num]);
1551.               object.endX = transfX(xVector[object.num+1]);
1552.               object.endY = transfY(solve[s1_cur][object.num+1]);
1553.
1554.               object.len = Math.sqrt(Math.pow((object.endY - object.y), 2) + Math.pow((object.endX -
    object.x), 2));
1555.
1556.               if (setupType == 1) {
1557.                   object.visible = true;
1558.               }
1559.               else {
1560.                   object.visible = false;
1561.               }
1562.
1563.               object.scaleX = object.len;
1564.               object.scaleY = 1;
1565.
1566.               object.rotation = Math.atan2((object.endY - object.y), (object.endX - object.x)) * 180
    / Math.PI;
1567.           }
```

```
1568.           else if (object.gen != generation) {
1569.               object.alpha -= 3/30;
1570.           }
1571.
1572.           if (object.alpha <= 0) {
1573.               object.alpha = 0;
1574.               object.visible = false;
1575.               object.removeEventListener('tick', setPoses21);
1576.               stage.removeChild(object);
1577.           }
1578.
1579.       }
1580.
1581.       function setPoses22(e) {
1582.           var object = e.currentTarget;
1583.           if (object.gen == generation) {
1584.               object.x = transfX(yVector[object.num]);
1585.               object.y = transfY(solve[object.num][s2_cur]);
1586.               object.endX = transfX(yVector[object.num+1]);
1587.               object.endY = transfY(solve[object.num+1][s2_cur]);
1588.
1589.               object.len = Math.sqrt(Math.pow((object.endY - object.y), 2) + Math.pow((object.endX -
    object.x), 2));
1590.
1591.               if (setupType == 2) {
1592.                   object.visible = true;
1593.               }
1594.               else {
1595.                   object.visible = false;
1596.               }
1597.
1598.
1599.               object.scaleX = object.len;
1600.               object.scaleY = 1;
1601.
1602.               object.rotation = Math.atan2((object.endY - object.y), (object.endX - object.x)) * 180
    / Math.PI;
1603.           }
1604.           else if (object.gen != generation) {
1605.               object.alpha -= 3/30;
1606.           }
1607.
1608.           if (object.alpha <= 0) {
1609.               object.alpha = 0;
1610.               object.visible = false;
1611.               object.removeEventListener('tick', setPoses22);
1612.               stage.removeChild(object);
1613.           }
1614.
1615.       }
1616.
1617.
1618.       function setPoses31(e) {
1619.           var object = e.currentTarget;
1620.           if (object.gen == generation) {
1621.               object.x = transfEX(xVector[object.num]);
1622.               object.y = transfEY(errorGraphX[s1_cur][object.num]);
1623.               object.endX = transfEX(xVector[object.num+1]);
1624.               object.endY = transfEY(errorGraphX[s1_cur][object.num+1]);
1625.
1626.               object.len = Math.sqrt(Math.pow((object.endY - object.y), 2) + Math.pow((object.endX -
    object.x), 2));
1627.
1628.               object.scaleX = object.len;
1629.               object.scaleY = 1;
1630.
1631.               object.rotation = Math.atan2((object.endY - object.y), (object.endX - object.x)) * 180
    / Math.PI;
```

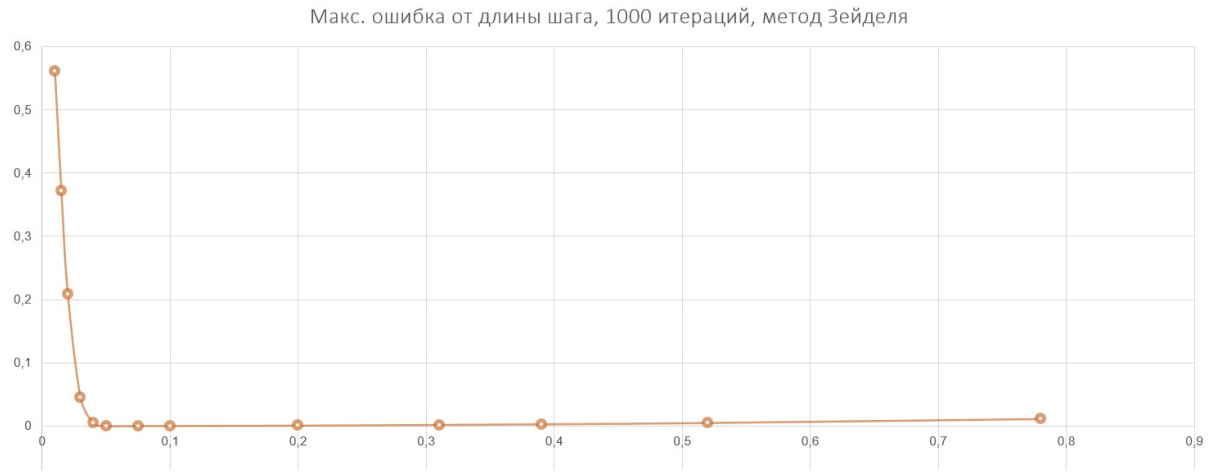
```
1632.           }
1633.           else if (object.gen != generation) {
1634.               object.alpha -= 3/30;
1635.           }
1636.
1637.           if (pogr_type <= 2) {
1638.               if (setupType == 1) {
1639.                   object.visible = true;
1640.               }
1641.               else {
1642.                   object.visible = false;
1643.               }
1644.           }
1645.           else {
1646.               object.visible = false;
1647.           }
1648.
1649.           if (object.alpha <= 0) {
1650.               object.alpha = 0;
1651.               object.visible = false;
1652.               object.removeEventListener('tick', setPoses31);
1653.               stage.removeChild(object);
1654.           }
1655.
1656.       }
1657.
1658.
1659.       function setPoses32(e) {
1660.           var object = e.currentTarget;
1661.           if (object.gen == generation) {
1662.               object.x = transfEX(yVector[object.num]);
1663.               object.y = transfEY(errorGraphY[s2_cur][object.num]);
1664.               //object.y = transfEY(errorGraph[object.num][s2_cur]);
1665.               object.endX = transfEX(yVector[object.num+1]);
1666.               object.endY = transfEY(errorGraphY[s2_cur][object.num+1]);
1667.               //object.endY = transfEY(errorGraph[object.num+1][s2_cur]);
1668.
1669.               object.len = Math.sqrt(Math.pow((object.endY - object.y), 2) + Math.pow((object.endX -
    object.x), 2));
1670.
1671.
1672.
1673.               object.scaleX = object.len;
1674.               object.scaleY = 1;
1675.
1676.               object.rotation = Math.atan2((object.endY - object.y), (object.endX - object.x)) * 180
    / Math.PI;
1677.           }
1678.           else if (object.gen != generation) {
1679.               object.alpha -= 3/30;
1680.           }
1681.
1682.           if (pogr_type <= 2) {
1683.               if (setupType == 2) {
1684.                   object.visible = true;
1685.               }
1686.               else {
1687.                   object.visible = false;
1688.               }
1689.           }
1690.           else {
1691.               object.visible = false;
1692.           }
1693.
1694.           if (object.alpha <= 0) {
1695.               object.alpha = 0;
1696.               object.visible = false;
1697.               object.removeEventListener('tick', setPoses32);
```

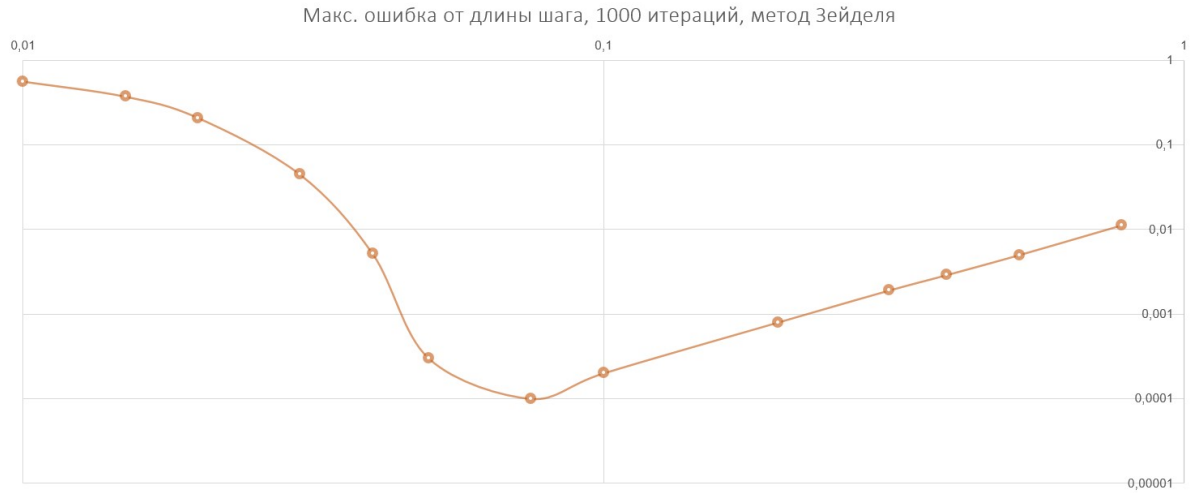
```
1698.               stage.removeChild(object);
1699.           }
1700.
1701.       }
1702.
1703.
1704.       function setPoses41(e) {
1705.           var object = e.currentTarget;
1706.           if (object.gen == generation) {
1707.               object.x = transfE2X(yVector[object.num]);
1708.               object.y = transfE2Y(EotTx[object.num]);
1709.               object.endX = transfE2X(yVector[object.num+1]);
1710.               object.endY = transfE2Y(EotTx[object.num+1]);
1711.
1712.               object.len = Math.sqrt(Math.pow((object.endY - object.y), 2) + Math.pow((object.endX -
    object.x), 2));
1713.
1714.               if (setupType == 1) {
1715.                   object.visible = true;
1716.               }
1717.               else {
1718.                   object.visible = false;
1719.               }
1720.
1721.               object.scaleX = object.len;
1722.               object.scaleY = 1;
1723.
1724.               object.rotation = Math.atan2((object.endY - object.y), (object.endX - object.x)) * 180
    / Math.PI;
1725.           }
1726.           else if (object.gen != generation) {
1727.               object.alpha -= 3/30;
1728.           }
1729.
1730.           if (pogr_type == 3) {
1731.               if (setupType == 1) {
1732.                   object.visible = true;
1733.               }
1734.               else {
1735.                   object.visible = false;
1736.               }
1737.           }
1738.           else {
1739.               object.visible = false;
1740.           }
1741.
1742.           if (object.alpha <= 0) {
1743.               object.alpha = 0;
1744.               object.visible = false;
1745.               object.removeEventListener('tick', setPoses41);
1746.               stage.removeChild(object);
1747.           }
1748.
1749.       }
1750.
1751.
1752.
1753.       function setPoses42(e) {
1754.           var object = e.currentTarget;
1755.           if (object.gen == generation) {
1756.               object.x = transfE2X(xVector[object.num]);
1757.               object.y = transfE2Y(EotTy[object.num]);
1758.               object.endX = transfE2X(xVector[object.num+1]);
1759.               object.endY = transfE2Y(EotTy[object.num+1]);
1760.
1761.               object.len = Math.sqrt(Math.pow((object.endY - object.y), 2) + Math.pow((object.endX -
    object.x), 2));
1762.
```

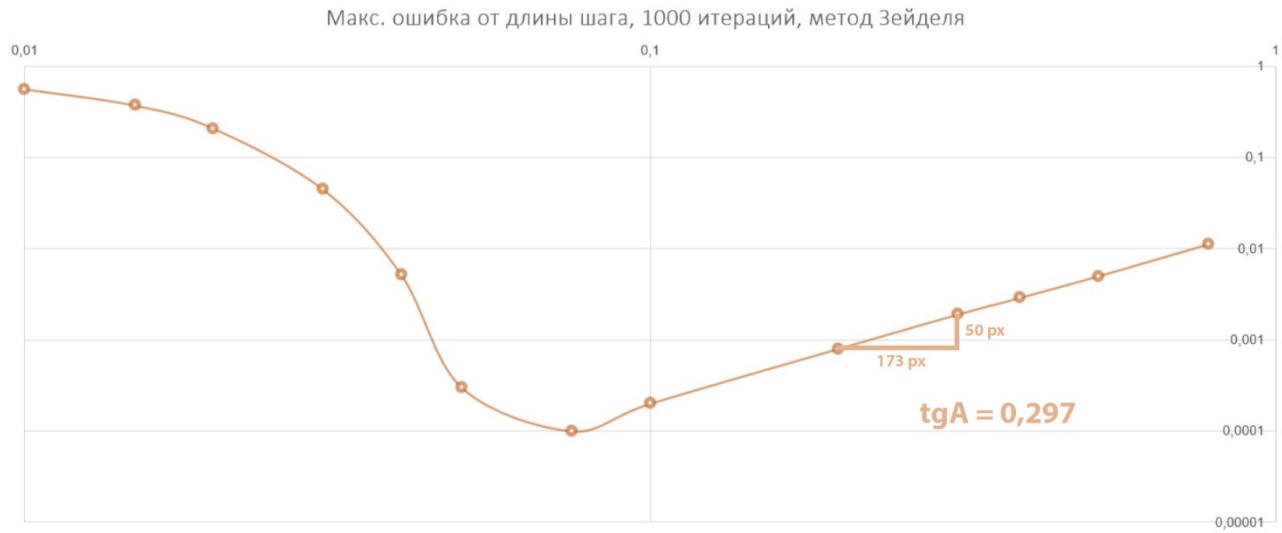
```
1763.
1764.
1765.                 object.scaleX = object.len;
1766.                 object.scaleY = 1;
1767.
1768.                 object.rotation = Math.atan2((object.endY - object.y), (object.endX - object.x)) * 180
    / Math.PI;
1769.             }
1770.             else if (object.gen != generation) {
1771.                 object.alpha -= 3/30;
1772.             }
1773.
1774.             if (pogr_type == 3) {
1775.                 if (setupType == 2) {
1776.                     object.visible = true;
1777.                 }
1778.                 else {
1779.                     object.visible = false;
1780.                 }
1781.             }
1782.             else {
1783.                 object.visible = false;
1784.             }
1785.
1786.             if (object.alpha <= 0) {
1787.                 object.alpha = 0;
1788.                 object.visible = false;
1789.                 object.removeEventListener('tick', setPoses42);
1790.                 stage.removeChild(object);
1791.             }
1792.
1793.         }
1794.
1795.
1796.
1797.
1798.
1799.
1800.
1801.         //метод прогонки
1802.         function progonka(matrix, vectorB) {
1803.             var vectorX = [];
1804.
1805.             var N = vectorB.length;
1806.
1807.             var alphas = [];
1808.             var betas = [];
1809.
1810.
1811.             for (var i = 0 ; i < vectorB.length ; ++i) {
1812.                 alphas.push(0);
1813.                 betas.push(0);
1814.             }
1815.
1816.             //Прямой ход прогонки
1817.
1818.             for (var i = 0; i < N; ++i) {
1819.                 var A0, C0, B0, F0;
1820.
1821.                 if (i - 1 < 0) {
1822.                     A0 = 0;
1823.                 }
1824.                 else {
1825.                     A0 = matrix[i][i - 1];
1826.                 }
1827.
1828.                 C0 = -1 * matrix[i][i];
1829.
```

```
1830.               if (i + 1 < N) {
1831.                   B0 = matrix[i][i + 1];
1832.               }
1833.               else {
1834.                   B0 = 0;
1835.               }
1836.
1837.               F0 = vectorB[i];
1838.
1839.
1840.               if (i == 0) {
1841.                   alphas[i] = B0 / C0;
1842.                   betas[i] = -(F0 / C0);
1843.               }
1844.               else if (i == N - 1) {
1845.                   alphas[i] = 0;
1846.                   betas[i] = (betas[i - 1] * A0 - F0) / (C0 - alphas[i - 1] * A0);
1847.               }
1848.               else {
1849.                   alphas[i] = B0 / (C0 - alphas[i - 1] * A0);
1850.                   betas[i] = (betas[i - 1] * A0 - F0) / (C0 - alphas[i - 1] * A0);
1851.               }
1852.
1853.           }
1854.
1855.
1856.           vectorX[N - 1] = betas[N - 1];
1857.
1858.           for (var i = 2; i <= N; ++i) {
1859.               vectorX[N - i] = alphas[N - i] * vectorX[N - i + 1] + betas[N - i];
1860.           }
1861.
1862.           return vectorX;
1863.       }
```

7. Графики







Ошибка в начале при маленьком шаге очень велика, потому что количество шагов очень большое, а 1000 итераций не успевают обработать данную задачу. Потом, когда количество шагов становится оптимальным, график достигает минимума. Затем снова растет, но уже из-за того, что шаги становятся больше. Также при разных количествах шагов по измерениям, ошибка довольно высока. Думаю, это связано с особенностью метода решения.

Данная лабораторная работа выполнена: **20 января 2021.**

Лабораторная работа 8

1. Тема ЛР:
   Метод конечных разностей решения многомерных задач математической физики. Методы расщепления. Схемы дробных шагов и переменных направлений.
2. Вариант : **7**

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} - xy\sin t\,,$$
$$u(0, y, t) = 0,$$
$$u(1, y, t) = y\cos t,$$
$$u(x, 0, t) = 0,$$
$$u(x, 1, t) = x\cos t,$$
$$u(x, y, 0) = xy\,.$$

Аналитическое решение: $U(x, y, t) = xy\cos t\,.$

3. Алгоритм:

Как по мне, так это самая сложная лабораторная из всех, примерно, как курсовая. На нее ушло более 12 часов непрерывной работы и было написано почти 3 тысячи строк кода. Сама лабораторная не такая сложная на самом деле, как работа, которую пришлось мне проделать. Трехмерная сетка, выбор временных координат, куча графиков, все это мне пришлось

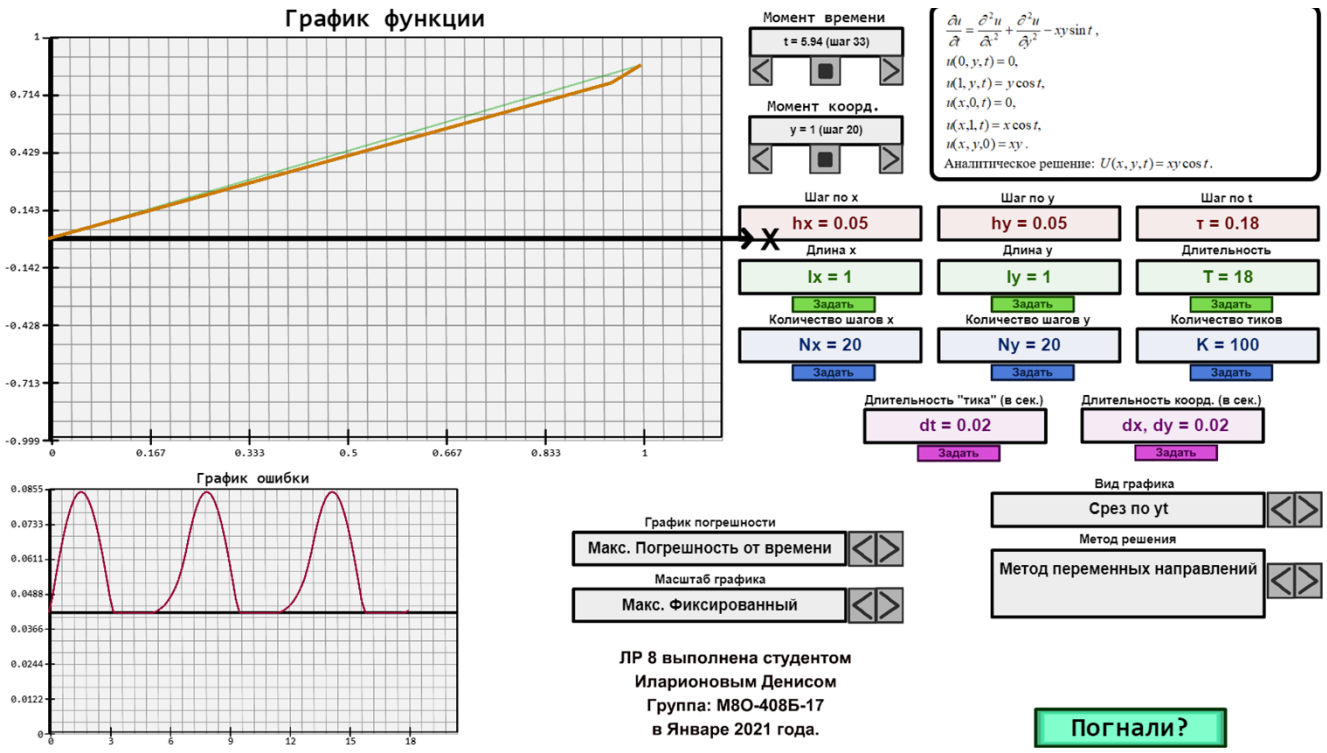
реализовать. Схемы тоже довольно сложные, из-за того, что я не рассматривал границы, они работали у меня некорректно. Но я вовремя это исправил, хотя и сидел над этим несколько часов. Обе схемы используют похожий принцип. Мы добавляем промежуточный временной шаг t + h/2. И в зависимости от четности шага выбираем по какой переменной у нас будет явная схема, а по какой неявная (в методе дробных шагов только неявная, более того, он работает даже лучше и дает меньше погрешность). С погрешностью тоже довольно трудно было, я решил переработать ее и сделать куб погрешностей и уже с него смотреть грани. Погрешность идет максимальная по обеим временным координатам при фиксированной точке.

4. Среда разработки:

Adobe Animate, Javascript + HTML5

5. Реализация

Выбор срезов, вид графика погрешностей. Убрал относительную погрешность, не вижу в ней смысла, да и работала она криво при значениях близких к нулю. На конце погрешность может быть равна нулю. Это связано с тем, что есть некоторые граничные условия. К счастью, они снова были заданы явно и мне не пришлось аппроксимировать их как в первой лабораторной. Но и без этого ушло немало усилий и трудов.



6. Код программы

```
1.    var alpha = 0; //α
```

```
2.    var beta = 1; //β
3.    var gamma = 0; //γ
4.    var delta = 1; //δ
5.
6.    //Обозначим все как граничные условия 3 рода
7.
8.    function phi_x0(y, t) {
9.        return 0;
10.   }
11.
12.   function phi_xl(y, t) {
13.       return y*Math.cos(t);
14.   }
15.
16.   function phi_y0(x, t) {
17.       return 0;
18.   }
19.
20.   function phi_yl(x, t) {
21.       return x*Math.cos(t);
22.   }
23.
24.   function phi_t0(x, y) {
25.       return x*y;
26.   }
27.
28.   function f(x, y, t) {
29.       return -1 * x * y * Math.sin(t);
30.   }
31.
32.
33.   function U(x, y, t) {
34.       return x*y*Math.cos(t);
35.   } //Аналитическое решение
36.
37.
38.   //Параметры сетки
39.   var lx = 1; //интервал x
40.   var Nx = 20; //количество шагов x
41.   var Ny = 20; //количество шагов y
42.   var K = 20; //количество тиков
43.   var ly = 1; //интервал y
44.   var T = 1; //длительность
45.
46.   var hx = 0; var hy = 0; var h = 0;
47.
48.   var a = 1; var b = 2; var c = -3;
49.
50.   var s1_cur = 0; //по y
51.   var s2_cur = 0; //по x
52.   var s3_cur = 0; //по t
53.
54.   var dt = 0.02; //длительность тика
55.   var dxy = 0.02; //длительность тика по кд
56.   var cur_tick = 0;
57.   var cur_tick2 = 0;
58.
59.
60.   var generation = 0; //номер генерации
61.   var go1 = false; //увеличивать счетчик тиков?
62.   var go2 = false; //увеличивать счетчик тиков?
63.
64.
65.   var setupType = 1;
66.   //1 срез по yt
67.   //2 срез по xt
68.
69.   var iters = 1000;
```

```
70.  //Количество итераций
71.
72.
73.  this.addEventListener("tick", det_ht.bind(this));
74.  function det_ht() {
75.      hx = lx / Nx;
76.      hy = ly / Ny;
77.      h = T / K;
78.
79.      //определяем шаги по времени и пространству
80.
81.      this.h1_text.text = "hx = " + Math.round(hx * 100000)/100000;
82.      this.h2_text.text = "hy = " + Math.round(hy * 100000)/100000;
83.      this.tau_text.text = "τ = " + Math.round(h * 100000)/100000;
84.
85.      this.l1_text.text = "lx = " + Math.round(lx * 100000)/100000;
86.      this.l2_text.text = "ly = " + Math.round(ly * 100000)/100000;
87.      this.T_text.text = "T = " + Math.round(T * 100000)/100000;
88.
89.      this.N1_text.text = "Nx = " + Math.round(Nx);
90.      this.N2_text.text = "Ny = " + Math.round(Ny);
91.      this.K_text.text = "K = " + Math.round(K);
92.
93.      this.dt_text.text = "dt = " + Math.round(dt*1000)/1000;
94.      this.dxy_text.text = "dx, dy = " + Math.round(dxy*1000)/1000;
95.
96.
97.      if (setupType == 1) {
98.          if (go1) {
99.              cur_tick += 1/60;
100.             if (cur_tick >= dxy && s1_cur < Ny) {
101.                 var tGot = Math.floor(cur_tick / dxy);
102.                 cur_tick %= dxy;
103.                 s1_cur += Math.min(Ny - s1_cur, tGot);
104.             }
105.             else if (s1_cur == Ny) {
106.                 go1 = false;
107.             }
108.         }
109.         if (go2) {
110.             cur_tick2 += 1/60;
111.             if (cur_tick2 >= dt && s3_cur < K) {
112.                 var tGot = Math.floor(cur_tick2 / dt);
113.                 cur_tick2 %= dt;
114.                 s3_cur += Math.min(K - s3_cur, tGot);
115.             }
116.             else if (s3_cur == K) {
117.                 go2 = false;
118.             }
119.         }
120.     }
121.     else if (setupType == 2) {
122.         if (go1) {
123.             cur_tick += 1/60;
124.             if (cur_tick >= dxy && s2_cur < Nx) {
125.                 var tGot = Math.floor(cur_tick / dxy);
126.                 cur_tick %= dxy;
127.                 s2_cur += Math.min(Nx - s2_cur, tGot);
128.             }
129.             else if (s2_cur == Nx) {
130.                 go1 = false;
131.             }
132.         }
133.         if (go2) {
134.             cur_tick2 += 1/60;
135.             if (cur_tick2 >= dt && s3_cur < K) {
136.                 var tGot = Math.floor(cur_tick2 / dt);
137.                 cur_tick2 %= dt;
```

```
                s3_cur += Math.min(K - s3_cur, tGot);
            }
            else if (s3_cur == K) {
                go2 = false;
            }
        }
    }
    else if (setupType == 3) {
        if (go1) {
            cur_tick += 1/60;
            if (cur_tick >= dxy && s2_cur < Nx) {
                var tGot = Math.floor(cur_tick / dxy);
                cur_tick %= dxy;
                s2_cur += Math.min(Nx - s2_cur, tGot);
            }
            else if (s2_cur == Nx) {
                go1 = false;
            }
        }
        if (go2) {
            cur_tick2 += 1/60;
            if (cur_tick2 >= dxy && s1_cur < Ny) {
                var tGot = Math.floor(cur_tick2 / dxy);
                cur_tick %= dxy;
                s1_cur += Math.min(Ny - s1_cur, tGot);
            }
            else if (s1_cur == Ny) {
                go2 = false;
            }
        }
    }

}



this.setGreen1.addEventListener("click", set_l1_prompt.bind(this));
function set_l1_prompt() {
    generation += 1;
    var temp = prompt("Введите lx:", '');
    temp = Number.parseFloat(temp);
    if (isNaN(temp)) {
        //если ввели какую-то хрень
        temp = lx;
    }
    else if (temp > 1000000) {
        //ограничение на величину
        temp = 1000000;
    }
    else if (temp <= 0) {
        //всегда положительно
        temp = 0.001;
    }
    lx = temp;
}


this.setGreen2.addEventListener("click", set_l2_prompt.bind(this));
function set_l2_prompt() {
    generation += 1;
    var temp = prompt("Введите ly:", '');
    temp = Number.parseFloat(temp);
    if (isNaN(temp)) {
        //если ввели какую-то хрень
        temp = ly;
    }
```

```
    else if (temp > 1000000) {
        //ограничение на величину
        temp = 1000000;
    }
    else if (temp <= 0) {
        //всегда положительно
        temp = 0.001;
    }
    ly = temp;
}


this.setGreen3.addEventListener("click", set_T_prompt.bind(this));
function set_T_prompt() {
    generation += 1;
    var temp = prompt("Введите T:", '');
    temp = Number.parseFloat(temp);
    if (isNaN(temp)) {
        //если ввели какую-то хрень
        temp = T;
    }
    else if (temp > 1000000) {
        //ограничение на величину
        temp = 1000000;
    }
    else if (temp <= 0) {
        //всегда положительно
        temp = 0.001;
    }
    T = temp;
}


this.setBlue1.addEventListener("click", set_Nx_prompt.bind(this));
function set_Nx_prompt() {
    generation += 1;
    var temp = prompt("Введите Nx:", '');
    temp = Number.parseInt(temp);
    if (isNaN(temp)) {
        //если ввели какую-то хрень
        temp = Nx;
    }
    else if (temp > 100) {
        //ограничение на величину
        temp = 100;
    }
    else if (temp <= 0) {
        //всегда положительно
        temp = 1;
    }
    Nx = temp;
}


this.setBlue2.addEventListener("click", set_Ny_prompt.bind(this));
function set_Ny_prompt() {
    generation += 1;
    var temp = prompt("Введите Ny:", '');
    temp = Number.parseInt(temp);
    if (isNaN(temp)) {
        //если ввели какую-то хрень
        temp = Ny;
    }
    else if (temp > 100) {
        //ограничение на величину
        temp = 100;
    }
```

```
    else if (temp <= 0) {
        //всегда положительно
        temp = 1;
    }
    Ny = temp;
}


this.setBlue3.addEventListener("click", set_K_prompt.bind(this));
function set_K_prompt() {
    generation += 1;
    var temp = prompt("Введите K:", '');
    temp = Number.parseInt(temp);
    if (isNaN(temp)) {
        //если ввели какую-то хрень
        temp = K;
    }
    else if (temp > 100) {
        //ограничение на величину
        temp = 100;
    }
    else if (temp <= 0) {
        //всегда положительно
        temp = 1;
    }
    K = temp;
}


this.setPurp1.addEventListener("click", set_dt_prompt.bind(this));
function set_dt_prompt() {
    cur_tick = 0;
    var temp = prompt("Введите dt (для ручного переключения можно ввести очень большим):", '');
    temp = Number.parseFloat(temp);
    if (isNaN(temp)) {
        //если ввели какую-то хрень
        temp = dt;
    }
    else if (temp > 1000000) {
        //ограничение на величину
        temp = 1000000;
    }
    else if (temp < 0.02) {
        //всегда положительно и больше 0.02 (почти плавная смена графика)
        temp = 0.02;
    }
    dt = temp;
}



this.setPurp2.addEventListener("click", set_dxy_prompt.bind(this));
function set_dxy_prompt() {
    cur_tick = 0;
    var temp = prompt("Введите dx и dy (для ручного переключения можно ввести очень большим):", '');
    temp = Number.parseFloat(temp);
    if (isNaN(temp)) {
        //если ввели какую-то хрень
        temp = dxy;
    }
    else if (temp > 1000000) {
        //ограничение на величину
        temp = 1000000;
    }
    else if (temp < 0.02) {
        //всегда положительно и больше 0.02 (почти плавная смена графика)
        temp = 0.02;
    }
```

```
342.     dxy = temp;
343. }
344.
345.
346.
347.
348.
349.
350. var meth_type = 1;
351. //Метод решения (схема)
352. //1 - метод переменных направлений
353. //2 - метод дробных шагов
354.
355.
356. var pogr_type = 1;
357. //Тип погрешности
358. //1 - Абсолютная
359. //2 - От времени 1
360. //3 - От времени 2
361.
362. var scale_type = 1;
363. //Масштаб графика
364. //1 - Фиксированный
365. //2 - Динамический
366.
367. this.SLeft1.addEventListener("click", scheme_left.bind(this));
368. function scheme_left() {
369.     if (setupType == 1) {
370.         setupType = 3;
371.     }
372.     else {
373.         setupType -= 1;
374.     }
375. }
376.
377. this.SRight1.addEventListener("click", scheme_right.bind(this));
378. function scheme_right() {
379.     if (setupType == 3) {
380.         setupType = 1;
381.     }
382.     else {
383.         setupType += 1;
384.     }
385. }
386.
387.
388. this.SLeft2.addEventListener("click", meth_left.bind(this));
389. function meth_left() {
390.     if (meth_type == 1) {
391.         meth_type = 2;
392.     }
393.     else {
394.         meth_type -= 1;
395.     }
396. }
397.
398. this.SRight2.addEventListener("click", meth_right.bind(this));
399. function meth_right() {
400.     if (meth_type == 2) {
401.         meth_type = 1;
402.     }
403.     else {
404.         meth_type += 1;
405.     }
406. }
407.
408.
409.
```

```
this.SLeft3.addEventListener("click", curStep_left.bind(this));
function curStep_left() {
    if (setupType <= 2) {
        if (s3_cur > 0) {
            s3_cur -= 1;
        }
    }
    else {
        if (s2_cur > 0) {
            s2_cur -= 1;
        }
    }
}



this.SLeft31.addEventListener("click", curStep_left2.bind(this));
function curStep_left2() {
    if (setupType == 1 || setupType == 3) {
        if (s1_cur > 0) {
            s1_cur -= 1;
        }
    }
    else {
        if (s2_cur > 0) {
            s2_cur -= 1;
        }
    }
}

this.SRight3.addEventListener("click", curStep_right.bind(this));
function curStep_right() {
    if (setupType <= 2) {
        if (s3_cur < K) {
            s3_cur += 1;
        }
    }
    else {
        if (s2_cur < Nx) {
            s2_cur += 1;
        }
    }
}

this.SRight31.addEventListener("click", curStep_right2.bind(this));
function curStep_right2() {
    if (setupType == 1 || setupType == 3) {
        if (s1_cur < Ny) {
            s1_cur += 1;
        }
    }
    else {
        if (s2_cur < Nx) {
            s2_cur += 1;
        }
    }
}


this.stopBtn.addEventListener("click", curStep_stop.bind(this));
function curStep_stop() {
    dt = 1000000;
}

this.SLeft4.addEventListener("click", pogr_left.bind(this));
function pogr_left() {
    if (pogr_type == 1) {
        pogr_type = 3;
```

```
    }
    else {
        pogr_type -= 1;
    }
}

this.SRight4.addEventListener("click", pogr_right.bind(this));
function pogr_right() {
    if (pogr_type == 3) {
        pogr_type = 1;
    }
    else {
        pogr_type += 1;
    }
}

this.SLeft5.addEventListener("click", scale_left.bind(this));
function scale_left() {
    if (scale_type == 1) {
        scale_type = 2;
    }
    else {
        scale_type -= 1;
    }
}

this.SRight5.addEventListener("click", scale_right.bind(this));
function scale_right() {
    if (scale_type == 2) {
        scale_type = 1;
    }
    else {
        scale_type += 1;
    }
}

this.addEventListener("tick", setTexts2.bind(this));
function setTexts2() {
    if (setupType == 1) {
        this.scheme_text.text = "Срез по yt";
    }
    else if (setupType == 2) {
        this.scheme_text.text = "Срез по xt";
    }
    else if (setupType == 3) {
        this.scheme_text.text = "Срез по xy";
    }

    if (meth_type == 1) {
        this.meth_text.text = "Метод переменных направлений";
    }
    else if (meth_type == 2) {
        this.meth_text.text = "Метод дробных шагов";
    }


    if (pogr_type == 1) {
        this.pogr_text.text = "Абсолютная погрешность";
    }
    else if (pogr_type == 2) {
        if (setupType <= 2) {
            this.pogr_text.text = "Макс. Погрешность от времени";
        }
        else {
            this.pogr_text.text = "Макс. Погрешность от X";
        }
    }
    else if (pogr_type == 3) {
```

```
546.         if (setupType == 1 || setupType == 3) {
547.             this.pogr_text.text = "Макс. Погрешность от Y";
548.         }
549.         else {
550.             this.pogr_text.text = "Макс. Погрешность от X";
551.         }
552.     }
553.
554.
555.     if (scale_type == 1) {
556.         this.scale_text.text = "Макс. Фиксированный";
557.     }
558.     else if (scale_type == 2) {
559.         this.scale_text.text = "Динамический";
560.     }
561.
562.
563.     if (setupType == 1) {
564.         this.divDown0.text = "" + 0;
565.         this.divDown1.text = "" + Math.round(lx*1/6*1000)/1000;
566.         this.divDown2.text = "" + Math.round(lx*2/6*1000)/1000;
567.         this.divDown3.text = "" + Math.round(lx*3/6*1000)/1000;
568.         this.divDown4.text = "" + Math.round(lx*4/6*1000)/1000;
569.         this.divDown5.text = "" + Math.round(lx*5/6*1000)/1000;
570.         this.divDown6.text = "" + Math.round(lx*1000)/1000;
571.     }
572.     else if (setupType == 2) {
573.         this.divDown0.text = "" + 0;
574.         this.divDown1.text = "" + Math.round(ly*1/6*1000)/1000;
575.         this.divDown2.text = "" + Math.round(ly*2/6*1000)/1000;
576.         this.divDown3.text = "" + Math.round(ly*3/6*1000)/1000;
577.         this.divDown4.text = "" + Math.round(ly*4/6*1000)/1000;
578.         this.divDown5.text = "" + Math.round(ly*5/6*1000)/1000;
579.         this.divDown6.text = "" + Math.round(ly*1000)/1000;
580.     }
581.     else if (setupType == 3) {
582.         this.divDown0.text = "" + 0;
583.         this.divDown1.text = "" + Math.round(T*1/6*1000)/1000;
584.         this.divDown2.text = "" + Math.round(T*2/6*1000)/1000;
585.         this.divDown3.text = "" + Math.round(T*3/6*1000)/1000;
586.         this.divDown4.text = "" + Math.round(T*4/6*1000)/1000;
587.         this.divDown5.text = "" + Math.round(T*5/6*1000)/1000;
588.         this.divDown6.text = "" + Math.round(T*1000)/1000;
589.     }
590.
591.     if (setupType == 1) {
592.         if (pogr_type == 1) {
593.             this.divDown0err.text = "" + 0;
594.             this.divDown1err.text = "" + Math.round(lx*1/6*1000)/1000;
595.             this.divDown2err.text = "" + Math.round(lx*2/6*1000)/1000;
596.             this.divDown3err.text = "" + Math.round(lx*3/6*1000)/1000;
597.             this.divDown4err.text = "" + Math.round(lx*4/6*1000)/1000;
598.             this.divDown5err.text = "" + Math.round(lx*5/6*1000)/1000;
599.             this.divDown6err.text = "" + Math.round(lx*1000)/1000;
600.         }
601.         else if (pogr_type == 2) {
602.             this.divDown0err.text = "" + 0;
603.             this.divDown1err.text = "" + Math.round(T*1/6*1000)/1000;
604.             this.divDown2err.text = "" + Math.round(T*2/6*1000)/1000;
605.             this.divDown3err.text = "" + Math.round(T*3/6*1000)/1000;
606.             this.divDown4err.text = "" + Math.round(T*4/6*1000)/1000;
607.             this.divDown5err.text = "" + Math.round(T*5/6*1000)/1000;
608.             this.divDown6err.text = "" + Math.round(T*1000)/1000;
609.         }
610.         else if (pogr_type == 3) {
611.             this.divDown0err.text = "" + 0;
612.             this.divDown1err.text = "" + Math.round(ly*1/6*1000)/1000;
613.             this.divDown2err.text = "" + Math.round(ly*2/6*1000)/1000;
```

```
614.             this.divDown3err.text = "" + Math.round(ly*3/6*1000)/1000;
615.             this.divDown4err.text = "" + Math.round(ly*4/6*1000)/1000;
616.             this.divDown5err.text = "" + Math.round(ly*5/6*1000)/1000;
617.             this.divDown6err.text = "" + Math.round(ly*1000)/1000;
618.         }
619.     }
620.     else if (setupType == 2) {
621.         if (pogr_type == 1) {
622.             this.divDown0err.text = "" + 0;
623.             this.divDown1err.text = "" + Math.round(ly*1/6*1000)/1000;
624.             this.divDown2err.text = "" + Math.round(ly*2/6*1000)/1000;
625.             this.divDown3err.text = "" + Math.round(ly*3/6*1000)/1000;
626.             this.divDown4err.text = "" + Math.round(ly*4/6*1000)/1000;
627.             this.divDown5err.text = "" + Math.round(ly*5/6*1000)/1000;
628.             this.divDown6err.text = "" + Math.round(ly*1000)/1000;
629.         }
630.         else if (pogr_type == 2) {
631.             this.divDown0err.text = "" + 0;
632.             this.divDown1err.text = "" + Math.round(T*1/6*1000)/1000;
633.             this.divDown2err.text = "" + Math.round(T*2/6*1000)/1000;
634.             this.divDown3err.text = "" + Math.round(T*3/6*1000)/1000;
635.             this.divDown4err.text = "" + Math.round(T*4/6*1000)/1000;
636.             this.divDown5err.text = "" + Math.round(T*5/6*1000)/1000;
637.             this.divDown6err.text = "" + Math.round(T*1000)/1000;
638.         }
639.         else if (pogr_type == 3) {
640.             this.divDown0err.text = "" + 0;
641.             this.divDown1err.text = "" + Math.round(lx*1/6*1000)/1000;
642.             this.divDown2err.text = "" + Math.round(lx*2/6*1000)/1000;
643.             this.divDown3err.text = "" + Math.round(lx*3/6*1000)/1000;
644.             this.divDown4err.text = "" + Math.round(lx*4/6*1000)/1000;
645.             this.divDown5err.text = "" + Math.round(lx*5/6*1000)/1000;
646.             this.divDown6err.text = "" + Math.round(lx*1000)/1000;
647.         }
648.     }
649.     else {
650.         if (pogr_type == 1) {
651.             this.divDown0err.text = "" + 0;
652.             this.divDown1err.text = "" + Math.round(T*1/6*1000)/1000;
653.             this.divDown2err.text = "" + Math.round(T*2/6*1000)/1000;
654.             this.divDown3err.text = "" + Math.round(T*3/6*1000)/1000;
655.             this.divDown4err.text = "" + Math.round(T*4/6*1000)/1000;
656.             this.divDown5err.text = "" + Math.round(T*5/6*1000)/1000;
657.             this.divDown6err.text = "" + Math.round(T*1000)/1000;
658.         }
659.         else if (pogr_type == 2) {
660.             this.divDown0err.text = "" + 0;
661.             this.divDown1err.text = "" + Math.round(lx*1/6*1000)/1000;
662.             this.divDown2err.text = "" + Math.round(lx*2/6*1000)/1000;
663.             this.divDown3err.text = "" + Math.round(lx*3/6*1000)/1000;
664.             this.divDown4err.text = "" + Math.round(lx*4/6*1000)/1000;
665.             this.divDown5err.text = "" + Math.round(lx*5/6*1000)/1000;
666.             this.divDown6err.text = "" + Math.round(lx*1000)/1000;
667.         }
668.         else if (pogr_type == 3) {
669.             this.divDown0err.text = "" + 0;
670.             this.divDown1err.text = "" + Math.round(ly*1/6*1000)/1000;
671.             this.divDown2err.text = "" + Math.round(ly*2/6*1000)/1000;
672.             this.divDown3err.text = "" + Math.round(ly*3/6*1000)/1000;
673.             this.divDown4err.text = "" + Math.round(ly*4/6*1000)/1000;
674.             this.divDown5err.text = "" + Math.round(ly*5/6*1000)/1000;
675.             this.divDown6err.text = "" + Math.round(ly*1000)/1000;
676.         }
677.     }
678.
679.     if (setupType == 1) {
680.         if (maxValsxt.length > 0 && scale_type == 2) {
681.             this.divUp0.text = "" + Math.round(minValsyt[s3_cur][s1_cur]*1000)/1000;
```

```
682.             this.divUp1.text = "" + Math.round((minValsyt[s3_cur][s1_cur] +
    (maxValsyt[s3_cur][s1_cur] - minValsyt[s3_cur][s1_cur])*1/7)*1000)/1000;
683.             this.divUp2.text = "" + Math.round((minValsyt[s3_cur][s1_cur] +
    (maxValsyt[s3_cur][s1_cur] - minValsyt[s3_cur][s1_cur])*2/7)*1000)/1000;
684.             this.divUp3.text = "" + Math.round((minValsyt[s3_cur][s1_cur] +
    (maxValsyt[s3_cur][s1_cur] - minValsyt[s3_cur][s1_cur])*3/7)*1000)/1000;
685.             this.divUp4.text = "" + Math.round((minValsyt[s3_cur][s1_cur] +
    (maxValsyt[s3_cur][s1_cur] - minValsyt[s3_cur][s1_cur])*4/7)*1000)/1000;
686.             this.divUp5.text = "" + Math.round((minValsyt[s3_cur][s1_cur] +
    (maxValsyt[s3_cur][s1_cur] - minValsyt[s3_cur][s1_cur])*5/7)*1000)/1000;
687.             this.divUp6.text = "" + Math.round((minValsyt[s3_cur][s1_cur] +
    (maxValsyt[s3_cur][s1_cur] - minValsyt[s3_cur][s1_cur])*6/7)*1000)/1000;
688.             this.divUp7.text = "" + Math.round(maxValsyt[s3_cur][s1_cur]*1000)/1000;
689.         }
690.         else {
691.             this.divUp0.text = "" + Math.round(minMinValyt*1000)/1000;
692.             this.divUp1.text = "" + Math.round((minMinValyt + (maxMaxValyt -
    minMinValyt)*1/7)*1000)/1000;
693.             this.divUp2.text = "" + Math.round((minMinValyt + (maxMaxValyt -
    minMinValyt)*2/7)*1000)/1000;
694.             this.divUp3.text = "" + Math.round((minMinValyt + (maxMaxValyt -
    minMinValyt)*3/7)*1000)/1000;
695.             this.divUp4.text = "" + Math.round((minMinValyt + (maxMaxValyt -
    minMinValyt)*4/7)*1000)/1000;
696.             this.divUp5.text = "" + Math.round((minMinValyt + (maxMaxValyt -
    minMinValyt)*5/7)*1000)/1000;
697.             this.divUp6.text = "" + Math.round((minMinValyt + (maxMaxValyt -
    minMinValyt)*6/7)*1000)/1000;
698.             this.divUp7.text = "" + Math.round(maxMaxValyt*1000)/1000;
699.         }
700.     }
701.     else if (setupType == 2) {
702.         if (maxValsyt.length > 0 && scale_type == 2) {
703.             this.divUp0.text = "" + Math.round(minValsxt[s3_cur][s2_cur]*1000)/1000;
704.             this.divUp1.text = "" + Math.round((minValsxt[s3_cur][s2_cur] +
    (maxValsxt[s3_cur][s2_cur] - minValsxt[s3_cur][s2_cur])*1/7)*1000)/1000;
705.             this.divUp2.text = "" + Math.round((minValsxt[s3_cur][s2_cur] +
    (maxValsxt[s3_cur][s2_cur] - minValsxt[s3_cur][s2_cur])*2/7)*1000)/1000;
706.             this.divUp3.text = "" + Math.round((minValsxt[s3_cur][s2_cur] +
    (maxValsxt[s3_cur][s2_cur] - minValsxt[s3_cur][s2_cur])*3/7)*1000)/1000;
707.             this.divUp4.text = "" + Math.round((minValsxt[s3_cur][s2_cur] +
    (maxValsxt[s3_cur][s2_cur] - minValsxt[s3_cur][s2_cur])*4/7)*1000)/1000;
708.             this.divUp5.text = "" + Math.round((minValsxt[s3_cur][s2_cur] +
    (maxValsxt[s3_cur][s2_cur] - minValsxt[s3_cur][s2_cur])*5/7)*1000)/1000;
709.             this.divUp6.text = "" + Math.round((minValsxt[s3_cur][s2_cur] +
    (maxValsxt[s3_cur][s2_cur] - minValsxt[s3_cur][s2_cur])*6/7)*1000)/1000;
710.             this.divUp7.text = "" + Math.round(maxValsxt[s3_cur][s2_cur]*1000)/1000;
711.         }
712.         else {
713.             this.divUp0.text = "" + Math.round(minMinValxt*1000)/1000;
714.             this.divUp1.text = "" + Math.round((minMinValxt + (maxMaxValxt -
    minMinValxt)*1/7)*1000)/1000;
715.             this.divUp2.text = "" + Math.round((minMinValxt + (maxMaxValxt -
    minMinValxt)*2/7)*1000)/1000;
716.             this.divUp3.text = "" + Math.round((minMinValxt + (maxMaxValxt -
    minMinValxt)*3/7)*1000)/1000;
717.             this.divUp4.text = "" + Math.round((minMinValxt + (maxMaxValxt -
    minMinValxt)*4/7)*1000)/1000;
718.             this.divUp5.text = "" + Math.round((minMinValxt + (maxMaxValxt -
    minMinValxt)*5/7)*1000)/1000;
719.             this.divUp6.text = "" + Math.round((minMinValxt + (maxMaxValxt -
    minMinValxt)*6/7)*1000)/1000;
720.             this.divUp7.text = "" + Math.round(maxMaxValxt*1000)/1000;
721.         }
722.     }
723.     else {
724.         if (maxValsxy.length > 0 && scale_type == 2) {
725.             this.divUp0.text = "" + Math.round(minValsxy[s1_cur][s2_cur]*1000)/1000;
```

```
726.             this.divUp1.text = "" + Math.round((minValsxy[s1_cur][s2_cur] +
    (maxValsxy[s1_cur][s2_cur] - minValsxy[s1_cur][s2_cur])*1/7)*1000)/1000;
727.             this.divUp2.text = "" + Math.round((minValsxy[s1_cur][s2_cur] +
    (maxValsxy[s1_cur][s2_cur] - minValsxy[s1_cur][s2_cur])*2/7)*1000)/1000;
728.             this.divUp3.text = "" + Math.round((minValsxy[s1_cur][s2_cur] +
    (maxValsxy[s1_cur][s2_cur] - minValsxy[s1_cur][s2_cur])*3/7)*1000)/1000;
729.             this.divUp4.text = "" + Math.round((minValsxy[s1_cur][s2_cur] +
    (maxValsxy[s1_cur][s2_cur] - minValsxy[s1_cur][s2_cur])*4/7)*1000)/1000;
730.             this.divUp5.text = "" + Math.round((minValsxy[s1_cur][s2_cur] +
    (maxValsxy[s1_cur][s2_cur] - minValsxy[s1_cur][s2_cur])*5/7)*1000)/1000;
731.             this.divUp6.text = "" + Math.round((minValsxy[s1_cur][s2_cur] +
    (maxValsxy[s1_cur][s2_cur] - minValsxy[s1_cur][s2_cur])*6/7)*1000)/1000;
732.             this.divUp7.text = "" + Math.round(maxValsxy[s1_cur][s2_cur]*1000)/1000;
733.         }
734.         else {
735.             this.divUp0.text = "" + Math.round(minMinValxy*1000)/1000;
736.             this.divUp1.text = "" + Math.round((minMinValxy + (maxMaxValxy -
    minMinValxy)*1/7)*1000)/1000;
737.             this.divUp2.text = "" + Math.round((minMinValxy + (maxMaxValxy -
    minMinValxy)*2/7)*1000)/1000;
738.             this.divUp3.text = "" + Math.round((minMinValxy + (maxMaxValxy -
    minMinValxy)*3/7)*1000)/1000;
739.             this.divUp4.text = "" + Math.round((minMinValxy + (maxMaxValxy -
    minMinValxy)*4/7)*1000)/1000;
740.             this.divUp5.text = "" + Math.round((minMinValxy + (maxMaxValxy -
    minMinValxy)*5/7)*1000)/1000;
741.             this.divUp6.text = "" + Math.round((minMinValxy + (maxMaxValxy -
    minMinValxy)*6/7)*1000)/1000;
742.             this.divUp7.text = "" + Math.round(maxMaxValxy*1000)/1000;
743.         }
744.     }
745.
746.     if (setupType == 1) {
747.         if (maxValsE.length > 0 && scale_type == 2 && pogr_type == 1) {
748.             this.divUp0err.text = "" + Math.round(minValsE[s3_cur][s1_cur]*1000)/1000;
749.             this.divUp1err.text = "" + Math.round((minValsE[s3_cur][s1_cur] +
    (maxValsE[s3_cur][s1_cur] - minValsE[s3_cur][s1_cur])*1/7)*1000)/1000;
750.             this.divUp2err.text = "" + Math.round((minValsE[s3_cur][s1_cur] +
    (maxValsE[s3_cur][s1_cur] - minValsE[s3_cur][s1_cur])*2/7)*1000)/1000;
751.             this.divUp3err.text = "" + Math.round((minValsE[s3_cur][s1_cur] +
    (maxValsE[s3_cur][s1_cur] - minValsE[s3_cur][s1_cur])*3/7)*1000)/1000;
752.             this.divUp4err.text = "" + Math.round((minValsE[s3_cur][s1_cur] +
    (maxValsE[s3_cur][s1_cur] - minValsE[s3_cur][s1_cur])*4/7)*1000)/1000;
753.             this.divUp5err.text = "" + Math.round((minValsE[s3_cur][s1_cur] +
    (maxValsE[s3_cur][s1_cur] - minValsE[s3_cur][s1_cur])*5/7)*1000)/1000;
754.             this.divUp6err.text = "" + Math.round((minValsE[s3_cur][s1_cur] +
    (maxValsE[s3_cur][s1_cur] - minValsE[s3_cur][s1_cur])*6/7)*1000)/1000;
755.             this.divUp7err.text = "" + Math.round(maxValsE[s3_cur][s1_cur]*1000)/1000;
756.         }
757.         else if (pogr_type == 1) {
758.             this.divUp0err.text = "" + Math.round(minMinValE*1000)/1000;
759.             this.divUp1err.text = "" + Math.round((minMinValE + (maxMaxValE -
    minMinValE)*1/7)*1000)/1000;
760.             this.divUp2err.text = "" + Math.round((minMinValE + (maxMaxValE -
    minMinValE)*2/7)*1000)/1000;
761.             this.divUp3err.text = "" + Math.round((minMinValE + (maxMaxValE -
    minMinValE)*3/7)*1000)/1000;
762.             this.divUp4err.text = "" + Math.round((minMinValE + (maxMaxValE -
    minMinValE)*4/7)*1000)/1000;
763.             this.divUp5err.text = "" + Math.round((minMinValE + (maxMaxValE -
    minMinValE)*5/7)*1000)/1000;
764.             this.divUp6err.text = "" + Math.round((minMinValE + (maxMaxValE -
    minMinValE)*6/7)*1000)/1000;
765.             this.divUp7err.text = "" + Math.round(maxMaxValE*1000)/1000;
766.         }
767.         else if (pogr_type == 2) {
768.             this.divUp0err.text = "" + Math.round(minEotTt*10000)/10000;
```

```
769.             this.divUp1err.text = "" + Math.round((minEotTt + (maxEotTt -
     minEotTt)*1/7)*10000)/10000;
770.             this.divUp2err.text = "" + Math.round((minEotTt + (maxEotTt -
     minEotTt)*2/7)*10000)/10000;
771.             this.divUp3err.text = "" + Math.round((minEotTt + (maxEotTt -
     minEotTt)*3/7)*10000)/10000;
772.             this.divUp4err.text = "" + Math.round((minEotTt + (maxEotTt -
     minEotTt)*4/7)*10000)/10000;
773.             this.divUp5err.text = "" + Math.round((minEotTt + (maxEotTt -
     minEotTt)*5/7)*10000)/10000;
774.             this.divUp6err.text = "" + Math.round((minEotTt + (maxEotTt -
     minEotTt)*6/7)*10000)/10000;
775.             this.divUp7err.text = "" + Math.round(maxEotTt*10000)/10000;
776.         }
777.         else if (pogr_type == 3) {
778.             this.divUp0err.text = "" + Math.round(minEotTy*10000)/10000;
779.             this.divUp1err.text = "" + Math.round((minEotTy + (maxEotTy -
     minEotTy)*1/7)*10000)/10000;
780.             this.divUp2err.text = "" + Math.round((minEotTy + (maxEotTy -
     minEotTy)*2/7)*10000)/10000;
781.             this.divUp3err.text = "" + Math.round((minEotTy + (maxEotTy -
     minEotTy)*3/7)*10000)/10000;
782.             this.divUp4err.text = "" + Math.round((minEotTy + (maxEotTy -
     minEotTy)*4/7)*10000)/10000;
783.             this.divUp5err.text = "" + Math.round((minEotTy + (maxEotTy -
     minEotTy)*5/7)*10000)/10000;
784.             this.divUp6err.text = "" + Math.round((minEotTy + (maxEotTy -
     minEotTy)*6/7)*10000)/10000;
785.             this.divUp7err.text = "" + Math.round(maxEotTy*10000)/10000;
786.         }
787.     }
788.     else if (setupType == 2) {
789.         if (maxValsE.length > 0 && scale_type == 2 && pogr_type == 1) {
790.             this.divUp0err.text = "" + Math.round(minValsE[s3_cur][s2_cur]*1000)/1000;
791.             this.divUp1err.text = "" + Math.round((minValsE[s3_cur][s2_cur] +
     (maxValsE[s3_cur][s2_cur] - minValsE[s3_cur][s2_cur])*1/7)*1000)/1000;
792.             this.divUp2err.text = "" + Math.round((minValsE[s3_cur][s2_cur] +
     (maxValsE[s3_cur][s2_cur] - minValsE[s3_cur][s2_cur])*2/7)*1000)/1000;
793.             this.divUp3err.text = "" + Math.round((minValsE[s3_cur][s2_cur] +
     (maxValsE[s3_cur][s2_cur] - minValsE[s3_cur][s2_cur])*3/7)*1000)/1000;
794.             this.divUp4err.text = "" + Math.round((minValsE[s3_cur][s2_cur] +
     (maxValsE[s3_cur][s2_cur] - minValsE[s3_cur][s2_cur])*4/7)*1000)/1000;
795.             this.divUp5err.text = "" + Math.round((minValsE[s3_cur][s2_cur] +
     (maxValsE[s3_cur][s2_cur] - minValsE[s3_cur][s2_cur])*5/7)*1000)/1000;
796.             this.divUp6err.text = "" + Math.round((minValsE[s3_cur][s2_cur] +
     (maxValsE[s3_cur][s2_cur] - minValsE[s3_cur][s2_cur])*6/7)*1000)/1000;
797.             this.divUp7err.text = "" + Math.round(maxValsE[s3_cur][s2_cur]*1000)/1000;
798.         }
799.         else if (pogr_type == 1) {
800.             this.divUp0err.text = "" + Math.round(minMinValE*1000)/1000;
801.             this.divUp1err.text = "" + Math.round((minMinValE + (maxMaxValE -
     minMinValE)*1/7)*1000)/1000;
802.             this.divUp2err.text = "" + Math.round((minMinValE + (maxMaxValE -
     minMinValE)*2/7)*1000)/1000;
803.             this.divUp3err.text = "" + Math.round((minMinValE + (maxMaxValE -
     minMinValE)*3/7)*1000)/1000;
804.             this.divUp4err.text = "" + Math.round((minMinValE + (maxMaxValE -
     minMinValE)*4/7)*1000)/1000;
805.             this.divUp5err.text = "" + Math.round((minMinValE + (maxMaxValE -
     minMinValE)*5/7)*1000)/1000;
806.             this.divUp6err.text = "" + Math.round((minMinValE + (maxMaxValE -
     minMinValE)*6/7)*1000)/1000;
807.             this.divUp7err.text = "" + Math.round(maxMaxValE*1000)/1000;
808.         }
809.         else if (pogr_type == 2) {
810.             this.divUp0err.text = "" + Math.round(minEotTt2*10000)/10000;
811.             this.divUp1err.text = "" + Math.round((minEotTt2 + (maxEotTt2 -
     minEotTt2)*1/7)*10000)/10000;
```

```
812.             this.divUp2err.text = "" + Math.round((minEotTt2 + (maxEotTt2 -
     minEotTt2)*2/7)*10000)/10000;
813.             this.divUp3err.text = "" + Math.round((minEotTt2 + (maxEotTt2 -
     minEotTt2)*3/7)*10000)/10000;
814.             this.divUp4err.text = "" + Math.round((minEotTt2 + (maxEotTt2 -
     minEotTt2)*4/7)*10000)/10000;
815.             this.divUp5err.text = "" + Math.round((minEotTt2 + (maxEotTt2 -
     minEotTt2)*5/7)*10000)/10000;
816.             this.divUp6err.text = "" + Math.round((minEotTt2 + (maxEotTt2 -
     minEotTt2)*6/7)*10000)/10000;
817.             this.divUp7err.text = "" + Math.round(maxEotTt2*10000)/10000;
818.         }
819.         else if (pogr_type == 3) {
820.             this.divUp0err.text = "" + Math.round(minEotTx2*10000)/10000;
821.             this.divUp1err.text = "" + Math.round((minEotTx2 + (maxEotTx2 -
     minEotTx2)*1/7)*10000)/10000;
822.             this.divUp2err.text = "" + Math.round((minEotTx2 + (maxEotTx2 -
     minEotTx2)*2/7)*10000)/10000;
823.             this.divUp3err.text = "" + Math.round((minEotTx2 + (maxEotTx2 -
     minEotTx2)*3/7)*10000)/10000;
824.             this.divUp4err.text = "" + Math.round((minEotTx2 + (maxEotTx2 -
     minEotTx2)*4/7)*10000)/10000;
825.             this.divUp5err.text = "" + Math.round((minEotTx2 + (maxEotTx2 -
     minEotTx2)*5/7)*10000)/10000;
826.             this.divUp6err.text = "" + Math.round((minEotTx2 + (maxEotTx2 -
     minEotTx2)*6/7)*10000)/10000;
827.             this.divUp7err.text = "" + Math.round(maxEotTx2*10000)/10000;
828.         }
829.     }
830.     else {
831.         if (maxValsE.length > 0 && scale_type == 2 && pogr_type == 1) {
832.             this.divUp0err.text = "" + Math.round(minValsE[s1_cur][s2_cur]*1000)/1000;
833.             this.divUp1err.text = "" + Math.round((minValsE[s1_cur][s2_cur] +
     (maxValsE[s1_cur][s2_cur] - minValsE[s1_cur][s2_cur])*1/7)*1000)/1000;
834.             this.divUp2err.text = "" + Math.round((minValsE[s1_cur][s2_cur] +
     (maxValsE[s1_cur][s2_cur] - minValsE[s1_cur][s2_cur])*2/7)*1000)/1000;
835.             this.divUp3err.text = "" + Math.round((minValsE[s1_cur][s2_cur] +
     (maxValsE[s1_cur][s2_cur] - minValsE[s1_cur][s2_cur])*3/7)*1000)/1000;
836.             this.divUp4err.text = "" + Math.round((minValsE[s1_cur][s2_cur] +
     (maxValsE[s1_cur][s2_cur] - minValsE[s1_cur][s2_cur])*4/7)*1000)/1000;
837.             this.divUp5err.text = "" + Math.round((minValsE[s1_cur][s2_cur] +
     (maxValsE[s1_cur][s2_cur] - minValsE[s1_cur][s2_cur])*5/7)*1000)/1000;
838.             this.divUp6err.text = "" + Math.round((minValsE[s1_cur][s2_cur] +
     (maxValsE[s1_cur][s2_cur] - minValsE[s1_cur][s2_cur])*6/7)*1000)/1000;
839.             this.divUp7err.text = "" + Math.round(maxValsE[s1_cur][s2_cur]*1000)/1000;
840.         }
841.         else if (pogr_type == 1) {
842.             this.divUp0err.text = "" + Math.round(minMinValE*1000)/1000;
843.             this.divUp1err.text = "" + Math.round((minMinValE + (maxMaxValE -
     minMinValE)*1/7)*1000)/1000;
844.             this.divUp2err.text = "" + Math.round((minMinValE + (maxMaxValE -
     minMinValE)*2/7)*1000)/1000;
845.             this.divUp3err.text = "" + Math.round((minMinValE + (maxMaxValE -
     minMinValE)*3/7)*1000)/1000;
846.             this.divUp4err.text = "" + Math.round((minMinValE + (maxMaxValE -
     minMinValE)*4/7)*1000)/1000;
847.             this.divUp5err.text = "" + Math.round((minMinValE + (maxMaxValE -
     minMinValE)*5/7)*1000)/1000;
848.             this.divUp6err.text = "" + Math.round((minMinValE + (maxMaxValE -
     minMinValE)*6/7)*1000)/1000;
849.             this.divUp7err.text = "" + Math.round(maxMaxValE*1000)/1000;
850.         }
851.         else if (pogr_type == 2) {
852.             this.divUp0err.text = "" + Math.round(minEotTx*10000)/10000;
853.             this.divUp1err.text = "" + Math.round((minEotTx + (maxEotTx -
     minEotTx)*1/7)*10000)/10000;
854.             this.divUp2err.text = "" + Math.round((minEotTx + (maxEotTx -
     minEotTx)*2/7)*10000)/10000;
```

```
            this.divUp3err.text = "" + Math.round((minEotTx + (maxEotTx - minEotTx)*3/7)*10000)/10000;
            this.divUp4err.text = "" + Math.round((minEotTx + (maxEotTx - minEotTx)*4/7)*10000)/10000;
            this.divUp5err.text = "" + Math.round((minEotTx + (maxEotTx - minEotTx)*5/7)*10000)/10000;
            this.divUp6err.text = "" + Math.round((minEotTx + (maxEotTx - minEotTx)*6/7)*10000)/10000;
            this.divUp7err.text = "" + Math.round(maxEotTx*10000)/10000;
        }
        else if (pogr_type == 3) {
            this.divUp0err.text = "" + Math.round(minEotTy2*10000)/10000;
            this.divUp1err.text = "" + Math.round((minEotTy2 + (maxEotTy2 - minEotTy2)*1/7)*10000)/10000;
            this.divUp2err.text = "" + Math.round((minEotTy2 + (maxEotTy2 - minEotTy2)*2/7)*10000)/10000;
            this.divUp3err.text = "" + Math.round((minEotTy2 + (maxEotTy2 - minEotTy2)*3/7)*10000)/10000;
            this.divUp4err.text = "" + Math.round((minEotTy2 + (maxEotTy2 - minEotTy2)*4/7)*10000)/10000;
            this.divUp5err.text = "" + Math.round((minEotTy2 + (maxEotTy2 - minEotTy2)*5/7)*10000)/10000;
            this.divUp6err.text = "" + Math.round((minEotTy2 + (maxEotTy2 - minEotTy2)*6/7)*10000)/10000;
            this.divUp7err.text = "" + Math.round(maxEotTy2*10000)/10000;
        }
    }


    if (setupType == 1) {
        var s1_cur2 = s1_cur*hy;
        var s3_cur2 = s3_cur*h;
        this.step_text.text = "t = " + Math.round(s3_cur2*10000)/10000 + " (war " + Math.round(s3_cur) + ")";
        this.step_text2.text = "y = " + Math.round(s1_cur2*10000)/10000 + " (war " + Math.round(s1_cur) + ")";
        this.axisDown.axisTp.text = "X";
    }
    else if (setupType == 2) {
        var s2_cur2 = s2_cur*hx;
        var s3_cur2 = s3_cur*h;
        this.step_text.text = "t = " + Math.round(s3_cur2*10000)/10000 + " (war " + Math.round(s3_cur) + ")";
        this.step_text2.text = "x = " + Math.round(s2_cur2*10000)/10000 + " (war " + Math.round(s2_cur) + ")";
        this.axisDown.axisTp.text = "Y";
    }
    else if (setupType == 3) {
        var s1_cur2 = s1_cur*hy;
        var s2_cur2 = s2_cur*hx;
        this.step_text.text = "x = " + Math.round(s2_cur2*10000)/10000 + " (war " + Math.round(s2_cur) + ")";
        this.step_text2.text = "y = " + Math.round(s1_cur2*10000)/10000 + " (war " + Math.round(s1_cur) + ")";
        this.axisDown.axisTp.text = "t";
    }
    this.axisDown.y = Math.min(Math.max(transfY(0), 26.4), 418.6);


    this.axisDownErr.y = Math.min(Math.max(transfEY(0), 468), 700);
}


//baseGraph
//x = 0 : posX = 60
//y = 0 : posY = 418.6

//x = max : posX = 627
```

```
907. //y = max : posY = 26.4
908.
909. //errGraph
910. //x = 0 : posX = 53.6
911. //y = 0 : posY = 700
912.
913. //x = max : posX = 397.8
914. //y = max : posY = 468.4
915.
916.
917.
918.
919.
920. var s1_cur = 0;
921. //текущий шаг по y
922.
923. var s2_cur = 0;
924. //текущий шаг по x
925.
926. var s3_cur = 0;
927. //текущий шаг по t
928.
929.
930. var etalon = [];
931. var xVector = [];
932. var yVector = [];
933. var tVector = [];
934.
935. var errorCube = []
936.
937. var maxMaxValyt = 1;
938. var minMinValyt = 0;
939.
940. var maxMaxValxt = 1;
941. var minMinValxt = 0;
942.
943. var maxMaxValxy = 1;
944. var minMinValxy = 0;
945.
946. var maxMaxValE = 1; //link
947. var minMinValE = 0; //link
948.
949. var maxMaxValE1yt = 1;
950. var minMinValE1yt = 0;
951.
952. var maxMaxValE1xt = 1;
953. var minMinValE1xt = 0;
954.
955. var maxMaxValE1xy = 1;
956. var minMinValE1xy = 0;
957.
958. var maxValsyt = [];
959. var minValsyt = [];
960.
961. var maxValsxt = [];
962. var minValsxt = [];
963.
964. var maxValsxy = [];
965. var minValsxy = [];
966.
967. var maxValsE = []; //link
968. var minValsE = []; //link
969.
970. var maxValsE1yt = [];
971. var minValsE1yt = [];
972.
973. var maxValsE1xt = [];
974. var minValsE1xt = [];
```

```
975.
976. var maxValsE1xy = [];
977. var minValsE1xy = [];
978.
979.
980.
981. var EotTx = [];
982. var EotTy = [];
983. var EotTt = [];
984.
985. var EotTx2 = [];
986. var EotTy2 = [];
987. var EotTt2 = [];
988.
989. this.addEventListener("tick", setError.bind(this));
990. function setError() {
991.     if (setupType == 1) {
992.         maxMaxValE = maxMaxValE1yt;
993.         minMinValE = minMinValE1yt;
994.         maxValsE = maxValsE1yt;
995.         minValsE = minValsE1yt;
996.     }
997.     else if (setupType == 2) {
998.         maxMaxValE = maxMaxValE1xt;
999.         minMinValE = minMinValE1xt;
1000.        maxValsE = maxValsE1xt;
1001.        minValsE = minValsE1xt;
1002.
1003.    }
1004.    else if (setupType == 3) {
1005.        maxMaxValE = maxMaxValE1xy;
1006.        minMinValE = minMinValE1xy;
1007.        maxValsE = maxValsE1xy;
1008.        minValsE = minValsE1xy;
1009.    }
1010.}
1011.
1012.var solve = [];
1013.var solve2 = [];
1014.
1015.
1016.function copyV(vector) {
1017.    var newV = [];
1018.    for (var i = 0 ; i < vector.length ; ++i) {
1019.        newV.push(vector[i]);
1020.    }
1021.    return newV;
1022.}
1023.
1024.function copyM(matrix) {
1025.    var newM = [];
1026.    for (var i = 0 ; i < matrix.length ; ++i) {
1027.        newM.push(copyV(matrix[i]));
1028.    }
1029.    return newM;
1030.}
1031.
1032.function copyC(cube) {
1033.    var newC = [];
1034.    for (var i = 0 ; i < cube.length ; ++i) {
1035.        newC.push(copyM(cube[i]));
1036.    }
1037.    return newC;
1038.}
1039.
1040.
1041.var minEotTx = 0;
1042.var maxEotTx = 1;
```

```
1043.
1044. var minEotTy = 0;
1045. var maxEotTy = 1;
1046.
1047. var minEotTt = 0;
1048. var maxEotTt = 1;
1049.
1050. var minEotTx2 = 0;
1051. var maxEotTx2 = 1;
1052.
1053. var minEotTy2 = 0;
1054. var maxEotTy2 = 1;
1055.
1056. var minEotTt2 = 0;
1057. var maxEotTt2 = 1;
1058.
1059.
1060.
1061.
1062. this.beginBtn.addEventListener("click", beginSimulation.bind(this));
1063. function beginSimulation() {
1064.     generation += 1;
1065.     s1_cur = 0;
1066.     s2_cur = 0;
1067.     s3_cur = 0;
1068.     etalon = [];
1069.     solve = [];
1070.     solve2 = [];
1071.     maxMaxValyt = 1;
1072.     minMinValyt = 0;
1073.
1074.     maxMaxValxt = 1;
1075.     minMinValxt = 0;
1076.
1077.     maxMaxValxy = 1;
1078.     minMinValxy = 0;
1079.
1080.     maxValsyt = [];
1081.     minValsyt = [];
1082.
1083.     for (var t = 0; t <= K ; ++t) {
1084.         var line = [];
1085.         for (var y = 0 ; y <= Ny ; ++y) {
1086.             line.push(0);
1087.         }
1088.         maxValsyt.push(copyV(line));
1089.         minValsyt.push(copyV(line));
1090.     }
1091.
1092.     maxValsxt = [];
1093.     minValsxt = [];
1094.
1095.     for (var t = 0; t <= K ; ++t) {
1096.         var line = [];
1097.         for (var x = 0 ; x <= Nx ; ++x) {
1098.             line.push(0);
1099.         }
1100.         maxValsxt.push(copyV(line));
1101.         minValsxt.push(copyV(line));
1102.     }
1103.
1104.     maxValsxy = [];
1105.     minValsxy = [];
1106.
1107.     for (var y = 0; y <= Ny ; ++y) {
1108.         var line = [];
1109.         for (var x = 0 ; x <= Nx ; ++x) {
1110.             line.push(0);
```

```
        }
        maxValsxy.push(copyV(line));
        minValsxy.push(copyV(line));
    }

    xVector = [];
    yVector = [];
    tVector = [];

    for (var j = 0 ; j <= Nx ; ++j) {
        xVector.push(hx * j);
    }
    for (var j = 0 ; j <= Ny ; ++j) {
        yVector.push(hy * j);
    }
    for (var j = 0 ; j <= K ; ++j) {
        tVector.push(h * j);
    }




    for (var i = 0 ; i <= K ; ++i) {
        var timeMatrix = [];
        var this_t = tVector[i];

        var this_maxX = 0;
        var this_minX = 0;

        for (var j = 0 ; j <= Ny ; ++j) {
            var timeVector = [];
            var this_y = yVector[j];

            for (var k = 0 ; k <= Nx ; ++k) {
                var this_x = xVector[k];
                var func_res = U(this_x, this_y, this_t);

                if (k == 0) {
                    maxValsyt[i][j] = func_res;
                    minValsyt[i][j] = func_res;
                }
                else {
                    maxValsyt[i][j] = (func_res > maxValsyt[i][j] ? func_res : maxValsyt[i][j]);
                    minValsyt[i][j] = (func_res < minValsyt[i][j] ? func_res : minValsyt[i][j]);
                }


                if (j == 0) {
                    maxValsxt[i][k] = func_res;
                    minValsxt[i][k] = func_res;
                }
                else {
                    maxValsxt[i][k] = (func_res > maxValsxt[i][k] ? func_res : maxValsxt[i][k]);
                    minValsxt[i][k] = (func_res < minValsxt[i][k] ? func_res : minValsxt[i][k]);
                }


                if (i == 0) {
                    maxValsxy[j][k] = func_res;
                    minValsxy[j][k] = func_res;
                }
                else {
                    maxValsxy[j][k] = (func_res > maxValsxy[j][k] ? func_res : maxValsxy[j][k]);
                    minValsxy[j][k] = (func_res < minValsxy[j][k] ? func_res : minValsxy[j][k]);
                }


                timeVector.push(func_res);
```

```
            }

            timeMatrix.push(timeVector);

        }
        etalon.push(timeMatrix);
    }

    solve2 = [];
    var downGrid = [];//нижний слой
    var emptyGrid = [];//пустая сетка

    for (var y = 0 ; y <= Ny ; ++y) {
        var timeVector = [];
        var emptyVector = [];
        var this_y = yVector[y];
        for (var x = 0 ; x <= Nx ; ++x) {
            var this_x = xVector[x];
            var res = phi_t0(this_x, this_y);
            timeVector.push(res);
            emptyVector.push(0);
        }
        downGrid.push(timeVector);
        emptyGrid.push(emptyVector);
    }

    solve2.push(downGrid);

    //метод переменных направлений
    if (meth_type == 1) {
        for (var it = 1 ; it <= 2*K ; ++it) {
            solve2.push(copyM(emptyGrid));
            var curT = it*h/2;

            //обрабатываем граничные условия
            for (var y = 0 ; y <= Ny ; ++y) {
                var curY = y*hy;
                solve2[it][y][0] = phi_x0(curY, curT);
                solve2[it][y][Nx] = phi_xl(curY, curT);
            }

            for (var x = 0 ; x <= Nx ; ++x) {
                var curX = x*hx;
                solve2[it][0][x] = phi_y0(curX, curT);
                solve2[it][Ny][x] = phi_yl(curX, curT);
            }

            var itsMatrix = [];

            //нечетный шаг схемы переменных направлений

            if (it%2 == 1) {
                for (var j = 1 ; j < Ny ; ++j) {
                    var curY = j*hy;
                    var solveMatr = [];
                    var resVector = [];
                    var BVector = [];
                    for (var i = 0 ; i <= Nx ; ++i) {
                        var curX = i*hx;
                        var freeVector = [];

                        for (var ii = 0 ; ii <= Nx ; ++ii) {
                            if (i == 0) {
                                if (ii == 0) {
                                    freeVector.push(1);
                                }
                                else {
                                    freeVector.push(0);
```

```
                                }
                            }
                            else if (i < Nx) {
                                if (ii < i - 1) {
                                    freeVector.push(0);
                                }
                                else if (ii == i - 1) {
                                    var A = h / (2 * hx * hx);
                                    freeVector.push(A);
                                }
                                else if (ii == i) {
                                    var B = -1 - (h / (hx * hx));
                                    freeVector.push(B);
                                }
                                else if (ii == i + 1) {
                                    var C = h / (2 * hx * hx);
                                    freeVector.push(C);
                                }
                                else {
                                    freeVector.push(0);
                                }
                            }
                            else {
                                if (ii == Nx) {
                                    freeVector.push(1);
                                }
                                else {
                                    freeVector.push(0);
                                }
                            }
                        }
                        solveMatr.push(freeVector);

                        if (i == 0) {
                            BVector.push(phi_x0(curY, curT));
                        }
                        else if (i < Nx) {
                            var D = -h/(2*hy*hy)*(solve2[it-1][j+1][i]
                                - 2*solve2[it-1][j][i] + solve2[it-1][j-1][i])
                                - solve2[it-1][j][i] - (h/2)*f(curX, curY, curT);


                            BVector.push(D);
                        }
                        else {
                            BVector.push(phi_xl(curY, curT));
                        }
                    }

                    resVector = progonka(solveMatr, BVector);
                    itsMatrix.push(copyV(resVector));
                }


                for (var y = 0 ; y < Ny - 1 ; ++y) {
                    for (var x = 0 ; x <= Nx ; ++x) {
                        solve2[it][y+1][x] = itsMatrix[y][x];
                    }
                }
            }
            else {
                for (var j = 1 ; j < Nx ; ++j) {
                    var curX = j*hx;
                    var solveMatr = [];
                    var resVector = [];
                    var BVector = [];
                    for (var i = 0 ; i <= Ny ; ++i) {
                        var curY = i*hy;
```

```
                        var freeVector = [];

                        for (var ii = 0 ; ii <= Ny ; ++ii) {
                            if (i == 0) {
                                if (ii == 0) {
                                    freeVector.push(1);
                                }
                                else {
                                    freeVector.push(0);
                                }
                            }
                            else if (i < Ny) {
                                if (ii < i - 1) {
                                    freeVector.push(0);
                                }
                                else if (ii == i - 1) {
                                    var A = h / (2 * hy * hy);
                                    freeVector.push(A);
                                }
                                else if (ii == i) {
                                    var B = -1 - (h / (hy * hy));
                                    freeVector.push(B);
                                }
                                else if (ii == i + 1) {
                                    var C = h / (2 * hy * hy);
                                    freeVector.push(C);
                                }
                                else {
                                    freeVector.push(0);
                                }
                            }
                            else {
                                if (ii == Ny) {
                                    freeVector.push(1);
                                }
                                else {
                                    freeVector.push(0);
                                }
                            }
                        }
                        solveMatr.push(copyV(freeVector));

                        if (i == 0) {
                            BVector.push(phi_y0(curX, curT - h/2));
                        }
                        else if (i < Ny) {
                            var D = -h/(2*hx*hx)*(solve2[it-1][i][j+1]
                                - 2*solve2[it-1][i][j] + solve2[it-1][i][j-1])
                                - solve2[it-1][i][j] - (h/2)*f(curX, curY, curT - h/2);

                            BVector.push(D);
                        }
                        else {
                            BVector.push(phi_yl(curX, curT - h/2));
                        }
                    }

                    resVector = progonka(solveMatr, BVector);


                    itsMatrix.push(copyV(resVector));
                }



                for (var y = 0 ; y <= Ny ; ++y) {
                    for (var x = 0 ; x < Nx - 1 ; ++x) {
                        solve2[it][y][x+1] = itsMatrix[x][y];
```

```
                    }
                }
            }
        }
    }
    else {
        for (var it = 1 ; it <= 2*K ; ++it) {
            solve2.push(copyM(emptyGrid));
            var curT = it*h/2;

            //обрабатываем граничные условия
            for (var y = 0 ; y <= Ny ; ++y) {
                var curY = y*hy;
                solve2[it][y][0] = phi_x0(curY, curT);
                solve2[it][y][Nx] = phi_xl(curY, curT);
            }

            for (var x = 0 ; x <= Nx ; ++x) {
                var curX = x*hx;
                solve2[it][0][x] = phi_y0(curX, curT);
                solve2[it][Ny][x] = phi_yl(curX, curT);
            }

            var itsMatrix = [];

            //нечетный шаг схемы дробных шагов

            if (it%2 == 1) {
                for (var j = 1 ; j < Ny ; ++j) {
                    var curY = j*hy;
                    var solveMatr = [];
                    var resVector = [];
                    var BVector = [];
                    for (var i = 0 ; i <= Nx ; ++i) {
                        var curX = i*hx;
                        var freeVector = [];

                        for (var ii = 0 ; ii <= Nx ; ++ii) {
                            if (i == 0) {
                                if (ii == 0) {
                                    freeVector.push(1);
                                }
                                else {
                                    freeVector.push(0);
                                }
                            }
                            else if (i < Nx) {
                                if (ii < i - 1) {
                                    freeVector.push(0);
                                }
                                else if (ii == i - 1) {
                                    var A = h / (hx * hx);
                                    freeVector.push(A);
                                }
                                else if (ii == i) {
                                    var B = -1 - (2 * h / (hx * hx));
                                    freeVector.push(B);
                                }
                                else if (ii == i + 1) {
                                    var C = h / (hx * hx);
                                    freeVector.push(C);
                                }
                                else {
                                    freeVector.push(0);
                                }
                            }
                            else {
                                if (ii == Nx) {
```

```
                                    freeVector.push(1);
                                }
                                else {
                                    freeVector.push(0);
                                }
                            }
                        }
                        solveMatr.push(freeVector);

                        if (i == 0) {
                            BVector.push(phi_x0(curY, curT-h/2));
                        }
                        else if (i < Nx) {
                            var D = -h*f(curX, curY, curT-h/2)/2 - solve2[it-1][j][i];

                            BVector.push(D);
                        }
                        else {
                            BVector.push(phi_xl(curY, curT-h/2));
                        }
                    }

                    resVector = progonka(solveMatr, BVector);
                    itsMatrix.push(copyV(resVector));
                }


                for (var y = 0 ; y < Ny - 1 ; ++y) {
                    for (var x = 0 ; x <= Nx ; ++x) {
                        solve2[it][y+1][x] = itsMatrix[y][x];
                    }
                }
            }
            else {
                for (var j = 1 ; j < Nx ; ++j) {
                    var curX = j*hx;
                    var solveMatr = [];
                    var resVector = [];
                    var BVector = [];
                    for (var i = 0 ; i <= Ny ; ++i) {
                        var curY = i*hy;
                        var freeVector = [];

                        for (var ii = 0 ; ii <= Ny ; ++ii) {
                            if (i == 0) {
                                if (ii == 0) {
                                    freeVector.push(1);
                                }
                                else {
                                    freeVector.push(0);
                                }
                            }
                            else if (i < Ny) {
                                if (ii < i - 1) {
                                    freeVector.push(0);
                                }
                                else if (ii == i - 1) {
                                    var A = h / (hy * hy);
                                    freeVector.push(A);
                                }
                                else if (ii == i) {
                                    var B = -1 - (2*h / (hy * hy));
                                    freeVector.push(B);
                                }
                                else if (ii == i + 1) {
                                    var C = h / (hy * hy);
                                    freeVector.push(C);
                                }
```

```
                                else {
                                    freeVector.push(0);
                                }
                            }
                            else {
                                if (ii == Ny) {
                                    freeVector.push(1);
                                }
                                else {
                                    freeVector.push(0);
                                }
                            }
                        }
                        solveMatr.push(copyV(freeVector));

                        if (i == 0) {
                            BVector.push(phi_y0(curX, curT));
                        }
                        else if (i < Ny) {
                            var D = -h*f(curX, curY, curT)/2 - solve2[it-1][i][j];

                            BVector.push(D);
                        }
                        else {
                            BVector.push(phi_yl(curX, curT));
                        }
                    }

                    resVector = progonka(solveMatr, BVector);


                    itsMatrix.push(copyV(resVector));
                }



                for (var y = 0 ; y <= Ny ; ++y) {
                    for (var x = 0 ; x < Nx - 1 ; ++x) {
                        solve2[it][y][x+1] = itsMatrix[x][y];
                    }
                }
            }
        }
    }









    for (var m = 0 ; m <= 2*K ; m+=2) {
        solve.push(copyM(solve2[m]));
    }


    //update minMax
    for (var i = 0 ; i <= K ; ++i) {

        for (var j = 0 ; j <= Ny ; ++j) {

            for (var k = 0 ; k <= Nx ; ++k) {

                maxValsyt[i][j] = (solve[i][j][k] > maxValsyt[i][j] ? solve[i][j][k] :
    maxValsyt[i][j]);
                minValsyt[i][j] = (solve[i][j][k] < minValsyt[i][j] ? solve[i][j][k] :
    minValsyt[i][j]);
```

```
1585.
1586.
1587.                 maxValsxt[i][k] = (solve[i][j][k] > maxValsxt[i][k] ? solve[i][j][k] :
     maxValsxt[i][k]);
1588.                 minValsxt[i][k] = (solve[i][j][k] < minValsxt[i][k] ? solve[i][j][k] :
     minValsxt[i][k]);
1589.
1590.                 maxValsxy[j][k] = (solve[i][j][k] > maxValsxy[j][k] ? solve[i][j][k] :
     maxValsxy[j][k]);
1591.                 minValsxy[j][k] = (solve[i][j][k] < minValsxy[j][k] ? solve[i][j][k] :
     minValsxy[j][k]);
1592.
1593.             }
1594.         }
1595.     }
1596.
1597.
1598.
1599.     maxMaxValyt = maxValsyt[0][0];
1600.     minMinValyt = minValsyt[0][0];
1601.
1602.     for (var t = 0; t <= K ; ++t) {
1603.         for (var y = 0 ; y <= Ny ; ++y) {
1604.             maxMaxValyt = (maxValsyt[t][y] > maxMaxValyt ? maxValsyt[t][y] : maxMaxValyt);
1605.             minMinValyt = (minValsyt[t][y] < minMinValyt ? minValsyt[t][y] : minMinValyt);
1606.         }
1607.     }
1608.
1609.     maxMaxValxt = maxValsxt[0][0];
1610.     minMinValxt = minValsxt[0][0];
1611.
1612.     for (var t = 0; t <= K ; ++t) {
1613.         for (var x = 0 ; x <= Nx ; ++x) {
1614.             maxMaxValxt = (maxValsxt[t][x] > maxMaxValxt ? maxValsxt[t][x] : maxMaxValxt);
1615.             minMinValxt = (minValsxt[t][x] < minMinValxt ? minValsxt[t][x] : minMinValxt);
1616.         }
1617.     }
1618.
1619.
1620.     maxMaxValxy = maxValsxy[0][0];
1621.     minMinValxy = minValsxy[0][0];
1622.
1623.     for (var y = 0; y <= Ny ; ++y) {
1624.         for (var x = 0 ; x <= Nx ; ++x) {
1625.             maxMaxValxy = (maxValsxy[y][x] > maxMaxValxy ? maxValsxy[y][x] : maxMaxValxy);
1626.             minMinValxy = (minValsxy[y][x] < minMinValxy ? minValsxy[y][x] : minMinValxy);
1627.         }
1628.     }
1629.
1630.
1631.     // error
1632.
1633.     maxMaxValE1yt = 1;
1634.     minMinValE1yt = 0;
1635.
1636.     maxMaxValE1xt = 1;
1637.     minMinValE1xt = 0;
1638.
1639.     maxMaxValE1xy = 1;
1640.     minMinValE1xy = 0;
1641.
1642.     maxValsE1yt = [];
1643.     minValsE1yt = [];
1644.
1645.     maxValsE1xt = [];
1646.     minValsE1xt = [];
1647.
1648.     maxValsE1xy = [];
```

```
    minValsE1xy = [];

    EotTx = [];
    EotTy = [];
    EotTt = [];

    EotTx2 = [];
    EotTy2 = [];
    EotTt2 = [];


    errorCube = copyC(solve);

    for (var t = 0 ; t <= K ; ++t) {
        for (var y = 0 ; y <= Ny ; ++y) {
            for (var x = 0 ; x <= Nx ; ++x) {
                errorCube[t][y][x] -= etalon[t][y][x];
            }
        }
    }

    for (var t = 0 ; t <= K ; ++t) {
        maxValsE1yt.push([]);
        minValsE1yt.push([]);
        for (var y = 0 ; y <= Ny ; ++y) {
            for (var x = 0 ; x <= Nx ; ++x) {
                if (x == 0) {
                    maxValsE1yt[t].push(errorCube[t][y][x]);
                    minValsE1yt[t].push(errorCube[t][y][x]);
                }
                else {
                    maxValsE1yt[t][y] = (errorCube[t][y][x] > maxValsE1yt[t][y] ? errorCube[t][y][x]
    : maxValsE1yt[t][y]);
                    minValsE1yt[t][y] = (errorCube[t][y][x] < minValsE1yt[t][y] ? errorCube[t][y][x]
    : minValsE1yt[t][y]);
                }
            }
        }
    }

    maxMaxValE1yt = maxValsE1yt[0][0];
    minMinValE1yt = minValsE1yt[0][0];

    for (var t = 0; t <= K ; ++t) {
        for (var y = 0 ; y <= Ny ; ++y) {
            maxMaxValE1yt = (maxValsE1yt[t][y] > maxMaxValE1yt ? maxValsE1yt[t][y] : maxMaxValE1yt);
            minMinValE1yt = (minValsE1yt[t][y] < minMinValE1yt ? minValsE1yt[t][y] : minMinValE1yt);
        }
    }



    for (var t = 0 ; t <= K ; ++t) {
        maxValsE1xt.push([]);
        minValsE1xt.push([]);
        for (var x = 0 ; x <= Nx ; ++x) {
            for (var y = 0 ; y <= Ny ; ++y) {
                if (y == 0) {
                    maxValsE1xt[t].push(errorCube[t][y][x]);
                    minValsE1xt[t].push(errorCube[t][y][x]);
                }
                else {
                    maxValsE1xt[t][x] = (errorCube[t][y][x] > maxValsE1xt[t][x] ? errorCube[t][y][x]
    : maxValsE1xt[t][x]);
                    minValsE1xt[t][x] = (errorCube[t][y][x] < minValsE1xt[t][x] ? errorCube[t][y][x]
    : minValsE1xt[t][x]);
                }
            }
```

```
        }
    }

    maxMaxValE1xt = maxValsE1xt[0][0];
    minMinValE1xt = minValsE1xt[0][0];

    for (var t = 0; t <= K ; ++t) {
        for (var x = 0 ; x <= Nx ; ++x) {
            maxMaxValE1xt = (maxValsE1xt[t][x] > maxMaxValE1xt ? maxValsE1xt[t][x] : maxMaxValE1xt);
            minMinValE1xt = (minValsE1xt[t][x] < minMinValE1xt ? minValsE1xt[t][x] : minMinValE1xt);
        }
    }


    for (var y = 0 ; y <= Ny ; ++y) {
        maxValsE1xy.push([]);
        minValsE1xy.push([]);
        for (var x = 0 ; x <= Nx ; ++x) {
            for (var t = 0 ; t <= K ; ++t) {
                if (t == 0) {
                    maxValsE1xy[y].push(errorCube[t][y][x]);
                    minValsE1xy[y].push(errorCube[t][y][x]);
                }
                else {
                    maxValsE1xy[y][x] = (errorCube[t][y][x] > maxValsE1xy[y][x] ? errorCube[t][y][x]
    : maxValsE1xy[y][x]);
                    minValsE1xy[y][x] = (errorCube[t][y][x] < minValsE1xy[y][x] ? errorCube[t][y][x]
    : minValsE1xy[y][x]);
                }
            }
        }
    }

    maxMaxValE1xy = maxValsE1xy[0][0];
    minMinValE1xy = minValsE1xy[0][0];

    for (var y = 0; y <= Ny ; ++y) {
        for (var x = 0 ; x <= Nx ; ++x) {
            maxMaxValE1xy = (maxValsE1xy[y][x] > maxMaxValE1xy ? maxValsE1xy[y][x] : maxMaxValE1xy);
            minMinValE1xy = (minValsE1xy[y][x] < minMinValE1xy ? minValsE1xy[y][x] : minMinValE1xy);
        }
    }



    //eotT

    for (var x = 0 ; x <= Nx ; ++x) {
        for (var t = 0 ; t <= K ; ++t) {
            for (var y = 0 ; y <= Ny ; ++y) {
                if (y == 0) {
                    EotTx.push(errorCube[t][y][x]);
                }
                else {
                    EotTx[x] = (errorCube[t][y][x] > EotTx[x] ? errorCube[t][y][x] : EotTx[x]);
                }

                if (t == 0) {
                    EotTx2.push(errorCube[t][y][x]);
                }
                else {
                    EotTx2[x] = (errorCube[t][y][x] > EotTx2[x] ? errorCube[t][y][x] : EotTx2[x]);
                }
            }
        }
    }

    minEotTx = EotTx[0];
```

```
    maxEotTx = EotTx[0];

    minEotTx2 = EotTx2[0];
    maxEotTx2 = EotTx2[0];

    for (var x = 0 ; x <= Nx ; ++x) {
        minEotTx = (EotTx[x] < minEotTx ? EotTx[x] : minEotTx);
        minEotTx2 = (EotTx2[x] < minEotTx2 ? EotTx2[x] : minEotTx2);

        maxEotTx = (EotTx[x] > maxEotTx ? EotTx[x] : maxEotTx);
        maxEotTx2 = (EotTx2[x] > maxEotTx2 ? EotTx2[x] : maxEotTx2);
    }


    for (var y = 0 ; y <= Ny ; ++y) {
        for (var x = 0 ; x <= Nx ; ++x) {
            for (var t = 0 ; t <= K ; ++t) {
                if (t == 0) {
                    EotTy.push(errorCube[t][y][x]);
                }
                else {
                    EotTy[y] = (errorCube[t][y][x] > EotTy[y] ? errorCube[t][y][x] : EotTy[y]);
                }

                if (x == 0) {
                    EotTy2.push(errorCube[t][y][x]);
                }
                else {
                    EotTy2[y] = (errorCube[t][y][x] > EotTy2[y] ? errorCube[t][y][x] : EotTy2[y]);
                }
            }
        }
    }


    minEotTy = EotTy[0];
    maxEotTy = EotTy[0];

    minEotTy2 = EotTy2[0];
    maxEotTy2 = EotTy2[0];

    for (var y = 0 ; y <= Ny ; ++y) {
        minEotTy = (EotTy[y] < minEotTy ? EotTy[y] : minEotTy);
        minEotTy2 = (EotTy2[y] < minEotTy2 ? EotTy2[y] : minEotTy2);

        maxEotTy = (EotTy[y] > maxEotTy ? EotTy[y] : maxEotTy);
        maxEotTy2 = (EotTy2[y] > maxEotTy2 ? EotTy2[y] : maxEotTy2);
    }




    for (var t = 0 ; t <= K ; ++t) {
        for (var x = 0 ; x <= Nx ; ++x) {
            for (var y = 0 ; y <= Ny ; ++y) {
                if (y == 0) {
                    EotTt.push(errorCube[t][y][x]);
                }
                else {
                    EotTt[t] = (errorCube[t][y][x] > EotTt[t] ? errorCube[t][y][x] : EotTt[t]);
                }

                if (x == 0) {
                    EotTt2.push(errorCube[t][y][x]);
                }
                else {
                    EotTt2[t] = (errorCube[t][y][x] > EotTt2[t] ? errorCube[t][y][x] : EotTt2[t]);
                }
```

```
            }
        }
    }

    minEotTt = EotTt[0];
    maxEotTt = EotTt[0];

    minEotTt2 = EotTt2[0];
    maxEotTt2 = EotTt2[0];

    for (var t = 0 ; t <= K ; ++t) {
        minEotTt = (EotTt[t] < minEotTt ? EotTt[t] : minEotTt);
        minEotTt2 = (EotTt2[t] < minEotTt2 ? EotTt2[t] : minEotTt2);

        maxEotTt = (EotTt[t] > maxEotTt ? EotTt[t] : maxEotTt);
        maxEotTt2 = (EotTt2[t] > maxEotTt2 ? EotTt2[t] : maxEotTt2);
    }

    makeEtalonGraph();
    makeSolveGraph();
    makeErrorGraph();
    makeErrorTGraph();

    go1 = true;
    go2 = true;
}

function transfX(x) {
    var newX;
    if (setupType == 1) {
        newX = 60 + (567 * (x / lx));
    }
    else if (setupType == 2) {
        newX = 60 + (567 * (x / ly));
    }
    else {
        newX = 60 + (567 * (x / T));
    }
    return newX;
}

function transfY(y) {
    var newY = 0;
    if (setupType == 1) {
        if (maxValsyt.length > 0 && scale_type == 2) {
            newY = 418.6 - (382.6 * (y - minValsyt[s3_cur][s1_cur])/(maxValsyt[s3_cur][s1_cur] -
    minValsyt[s3_cur][s1_cur]));
        }
        else {
            newY = 418.6 - (382.6 * (y - minMinValyt)/(maxMaxValyt - minMinValyt));
        }
    }
    else if (setupType == 2) {
        if (maxValsxt.length > 0 && scale_type == 2) {
            newY = 418.6 - (382.6 * (y - minValsxt[s3_cur][s2_cur])/(maxValsxt[s3_cur][s2_cur] -
    minValsxt[s3_cur][s2_cur]));
        }
        else {
            newY = 418.6 - (382.6 * (y - minMinValxt)/(maxMaxValxt - minMinValxt));
        }
    }
    else if (setupType == 3) {
        if (maxValsxy.length > 0 && scale_type == 2) {
            newY = 418.6 - (382.6 * (y - minValsxy[s1_cur][s2_cur])/(maxValsxy[s1_cur][s2_cur] -
    minValsxy[s1_cur][s2_cur]));
        }
        else {
            newY = 418.6 - (382.6 * (y - minMinValxy)/(maxMaxValxy - minMinValxy));
```

```
        }
    }
    return newY;
}


function transfEX(x) {
    var newX;
    if (setupType == 1) {
        newX = 60.6 + (344.2 * (x / lx));
    }
    else if (setupType == 2) {
        newX = 60.6 + (344.2 * (x / ly));
    }
    else {
        newX = 60.6 + (344.2 * (x / T));
    }
    return newX;
}


function transfE2X(x) {
    var newX;
    if (pogr_type <= 2) {
        if (setupType == 1) {
            newX = 60.6 + (344.2 * (x / T));
        }
        else if (setupType == 2) {
            newX = 60.6 + (344.2 * (x / T));
        }
        else {
            newX = 60.6 + (344.2 * (x / lx));
        }
    }
    else {
        if (setupType == 1) {
            newX = 60.6 + (344.2 * (x / ly));
        }
        else if (setupType == 2) {
            newX = 60.6 + (344.2 * (x / lx));
        }
        else {
            newX = 60.6 + (344.2 * (x / ly));
        }
    }
    return newX;
}

function transfEY(y) {
    var newY = 0;
    if (setupType == 1) {
        if (maxValsE.length > 0 && scale_type == 2) {
            newY = 700 - (231 * (y - minValsE[s1_cur][s3_cur])/(maxValsE[s1_cur][s3_cur] -
    minValsE[s1_cur][s3_cur]));
        }
        else {
            newY = 700 - (231 * (y - minMinValE)/(maxMaxValE - minMinValE));
        }
    }
    else if (setupType == 2) {
        if (maxValsE.length > 0 && scale_type == 2) {
            newY = 700 - (231 * (y - minValsE[s2_cur][s3_cur])/(maxValsE[s2_cur][s3_cur] -
    minValsE[s2_cur][s3_cur]));
        }
        else {
            newY = 700 - (231 * (y - minMinValE)/(maxMaxValE - minMinValE));
        }
    }
```

```
    else if (setupType == 3) {
        if (maxValsE.length > 0 && scale_type == 2) {
            newY = 700 - (231 * (y - minValsE[s1_cur][s2_cur])/(maxValsE[s1_cur][s2_cur] - minValsE[s1_cur][s2_cur]));
        }
        else {
            newY = 700 - (231 * (y - minMinValE)/(maxMaxValE - minMinValE));
        }
    }
    return newY;
}

function transfE2Y(y) {
    var newY = 0;
    if (setupType == 1) {
        newY = 700 - (231 * (y - minEotTx)/(maxEotTx - minEotTx));
    }
    else if (setupType == 2) {
        newY = 700 - (231 * (y - minEotTy)/(maxEotTy - minEotTy));
    }
    else if (setupType == 3) {
        newY = 700 - (231 * (y - minEotTt)/(maxEotTt - minEotTt));
    }

    return newY;
}


function makeEtalonGraph() {
    for (var i = 0 ; i < Nx ; ++i) {
        var join = new lib.line();
        stage.addChild(join);
        join.x = transfX(xVector[i]);
        join.y = transfY(etalon[0][0][i]);
        join.endX = transfX(xVector[i+1]);
        join.endY = transfY(etalon[0][0][i+1]);

        join.gotoAndStop(0);
        join.num = i;

        join.len = Math.sqrt(Math.pow((join.endY - join.y), 2) + Math.pow((join.endX - join.x), 2));

        join.scaleX = join.len;
        join.scaleY = 1;

        join.rotation = Math.atan2((join.endY - join.y), (join.endX - join.x)) * 180 / Math.PI;

        join.gen = generation;


        join.visible = true;
        join.alpha = 1;

        join.addEventListener('tick', setPoses11);
    }
    for (var i = 0 ; i < Ny ; ++i) {
        var join = new lib.line();
        stage.addChild(join);
        join.x = transfX(yVector[i]);
        join.y = transfY(etalon[0][i][0]);
        join.endX = transfX(yVector[i+1]);
        join.endY = transfY(etalon[0][i+1][0]);

        join.gotoAndStop(0);
        join.num = i;

        join.len = Math.sqrt(Math.pow((join.endY - join.y), 2) + Math.pow((join.endX - join.x), 2));

```

```
        join.scaleX = join.len;
        join.scaleY = 1;

        join.rotation = Math.atan2((join.endY - join.y), (join.endX - join.x)) * 180 / Math.PI;

        join.gen = generation;


        join.visible = true;
        join.alpha = 1;

        join.addEventListener('tick', setPoses12);
    }
    for (var i = 0 ; i < K ; ++i) {
        var join = new lib.line();
        stage.addChild(join);
        join.x = transfX(tVector[i]);
        join.y = transfY(etalon[i][0][0]);
        join.endX = transfX(tVector[i+1]);
        join.endY = transfY(etalon[i+1][0][0]);

        join.gotoAndStop(0);
        join.num = i;

        join.len = Math.sqrt(Math.pow((join.endY - join.y), 2) + Math.pow((join.endX - join.x), 2));

        join.scaleX = join.len;
        join.scaleY = 1;

        join.rotation = Math.atan2((join.endY - join.y), (join.endX - join.x)) * 180 / Math.PI;

        join.gen = generation;


        join.visible = true;
        join.alpha = 1;

        join.addEventListener('tick', setPoses13);
    }

}

function makeSolveGraph() {
    for (var i = 0 ; i < Nx ; ++i) {
        var join = new lib.line();
        stage.addChild(join);
        join.x = transfX(xVector[i]);
        join.y = transfY(solve[0][0][i]);
        join.endX = transfX(xVector[i+1]);
        join.endY = transfY(solve[0][0][i+1]);

        join.gotoAndStop(1);
        join.num = i;

        join.len = Math.sqrt(Math.pow((join.endY - join.y), 2) + Math.pow((join.endX - join.x), 2));

        join.scaleX = join.len;
        join.scaleY = 1;

        join.rotation = Math.atan2((join.endY - join.y), (join.endX - join.x)) * 180 / Math.PI;

        join.gen = generation;


        join.visible = true;
        join.alpha = 1;

        join.addEventListener('tick', setPoses21);
```

```
    }
    for (var i = 0 ; i < Ny ; ++i) {
        var join = new lib.line();
        stage.addChild(join);
        join.x = transfX(yVector[i]);
        join.y = transfY(solve[0][i][0]);
        join.endX = transfX(yVector[i+1]);
        join.endY = transfY(solve[0][i+1][0]);

        join.gotoAndStop(1);
        join.num = i;

        join.len = Math.sqrt(Math.pow((join.endY - join.y), 2) + Math.pow((join.endX - join.x), 2));

        join.scaleX = join.len;
        join.scaleY = 1;

        join.rotation = Math.atan2((join.endY - join.y), (join.endX - join.x)) * 180 / Math.PI;

        join.gen = generation;


        join.visible = true;
        join.alpha = 1;

        join.addEventListener('tick', setPoses22);
    }
    for (var i = 0 ; i < K ; ++i) {
        var join = new lib.line();
        stage.addChild(join);
        join.x = transfX(tVector[i]);
        join.y = transfY(solve[i][0][0]);
        join.endX = transfX(tVector[i+1]);
        join.endY = transfY(solve[i+1][0][0]);

        join.gotoAndStop(1);
        join.num = i;

        join.len = Math.sqrt(Math.pow((join.endY - join.y), 2) + Math.pow((join.endX - join.x), 2));

        join.scaleX = join.len;
        join.scaleY = 1;

        join.rotation = Math.atan2((join.endY - join.y), (join.endX - join.x)) * 180 / Math.PI;

        join.gen = generation;


        join.visible = true;
        join.alpha = 1;

        join.addEventListener('tick', setPoses23);
    }
}

function makeErrorGraph() {
    for (var i = 0 ; i < Nx ; ++i) {
        var join = new lib.line();
        stage.addChild(join);
        join.x = transfEX(xVector[i]);
        join.y = transfEY(errorCube[0][0][i]);
        join.endX = transfEX(xVector[i+1]);
        join.endY = transfEY(errorCube[0][0][i+1]);

        join.gotoAndStop(3);
        join.num = i;

        join.len = Math.sqrt(Math.pow((join.endY - join.y), 2) + Math.pow((join.endX - join.x), 2));
```

```
2181.
2182.         join.scaleX = join.len;
2183.         join.scaleY = 1;
2184.
2185.         join.rotation = Math.atan2((join.endY - join.y), (join.endX - join.x)) * 180 / Math.PI;
2186.
2187.         join.gen = generation;
2188.
2189.
2190.         join.visible = true;
2191.         join.alpha = 1;
2192.
2193.         join.addEventListener('tick', setPoses31);
2194.     }
2195.     for (var i = 0 ; i < Ny ; ++i) {
2196.         var join = new lib.line();
2197.         stage.addChild(join);
2198.         join.x = transfEX(yVector[i]);
2199.         join.y = transfEY(errorCube[0][i][0]);
2200.         join.endX = transfEX(yVector[i+1]);
2201.         join.endY = transfEY(errorCube[0][i+1][0]);
2202.
2203.         join.gotoAndStop(3);
2204.         join.num = i;
2205.
2206.         join.len = Math.sqrt(Math.pow((join.endY - join.y), 2) + Math.pow((join.endX - join.x), 2));
2207.
2208.         join.scaleX = join.len;
2209.         join.scaleY = 1;
2210.
2211.         join.rotation = Math.atan2((join.endY - join.y), (join.endX - join.x)) * 180 / Math.PI;
2212.
2213.         join.gen = generation;
2214.
2215.
2216.         join.visible = true;
2217.         join.alpha = 1;
2218.
2219.         join.addEventListener('tick', setPoses32);
2220.     }
2221.     for (var i = 0 ; i < K ; ++i) {
2222.         var join = new lib.line();
2223.         stage.addChild(join);
2224.         join.x = transfEX(tVector[i]);
2225.         join.y = transfEY(errorCube[i][0][0]);
2226.         join.endX = transfEX(tVector[i+1]);
2227.         join.endY = transfEY(errorCube[i+1][0][0]);
2228.
2229.         join.gotoAndStop(3);
2230.         join.num = i;
2231.
2232.         join.len = Math.sqrt(Math.pow((join.endY - join.y), 2) + Math.pow((join.endX - join.x), 2));
2233.
2234.         join.scaleX = join.len;
2235.         join.scaleY = 1;
2236.
2237.         join.rotation = Math.atan2((join.endY - join.y), (join.endX - join.x)) * 180 / Math.PI;
2238.
2239.         join.gen = generation;
2240.
2241.
2242.         join.visible = true;
2243.         join.alpha = 1;
2244.
2245.         join.addEventListener('tick', setPoses33);
2246.     }
2247.
2248.
```

```
}


function makeErrorTGraph() {
    for (var i = 0 ; i < Nx ; ++i) {
        var join = new lib.line();
        stage.addChild(join);
        join.x = transfE2X(xVector[i]);
        join.y = transfE2Y(EotTx[i]);
        join.endX = transfE2X(xVector[i+1]);
        join.endY = transfE2Y(EotTx[i+1]);

        join.gotoAndStop(3);
        join.num = i;

        join.len = Math.sqrt(Math.pow((join.endY - join.y), 2) + Math.pow((join.endX - join.x), 2));

        join.scaleX = join.len;
        join.scaleY = 1;

        join.rotation = Math.atan2((join.endY - join.y), (join.endX - join.x)) * 180 / Math.PI;

        join.gen = generation;


        join.visible = true;
        join.alpha = 1;

        join.addEventListener('tick', setPoses41);
    }
    for (var i = 0 ; i < Ny ; ++i) {
        var join = new lib.line();
        stage.addChild(join);
        join.x = transfE2X(yVector[i]);
        join.y = transfE2Y(EotTy[i]);
        join.endX = transfE2X(yVector[i+1]);
        join.endY = transfE2Y(EotTy[i+1]);

        join.gotoAndStop(3);
        join.num = i;

        join.len = Math.sqrt(Math.pow((join.endY - join.y), 2) + Math.pow((join.endX - join.x), 2));

        join.scaleX = join.len;
        join.scaleY = 1;

        join.rotation = Math.atan2((join.endY - join.y), (join.endX - join.x)) * 180 / Math.PI;

        join.gen = generation;


        join.visible = true;
        join.alpha = 1;

        join.addEventListener('tick', setPoses42);
    }
    for (var i = 0 ; i < K ; ++i) {
        var join = new lib.line();
        stage.addChild(join);
        join.x = transfE2X(tVector[i]);
        join.y = transfE2Y(EotTt[i]);
        join.endX = transfE2X(tVector[i+1]);
        join.endY = transfE2Y(EotTt[i+1]);

        join.gotoAndStop(3);
        join.num = i;

        join.len = Math.sqrt(Math.pow((join.endY - join.y), 2) + Math.pow((join.endX - join.x), 2));
```

```
2317.
2318.         join.scaleX = join.len;
2319.         join.scaleY = 1;
2320.
2321.         join.rotation = Math.atan2((join.endY - join.y), (join.endX - join.x)) * 180 / Math.PI;
2322.
2323.         join.gen = generation;
2324.
2325.
2326.         join.visible = true;
2327.         join.alpha = 1;
2328.
2329.         join.addEventListener('tick', setPoses43);
2330.     }
2331. }
2332.
2333.
2334.
2335. function setPoses11(e) {
2336.     var object = e.currentTarget;
2337.     if (object.gen == generation) {
2338.         object.x = transfX(xVector[object.num]);
2339.         object.y = transfY(etalon[s3_cur][s1_cur][object.num]);
2340.         object.endX = transfX(xVector[object.num+1]);
2341.         object.endY = transfY(etalon[s3_cur][s1_cur][object.num+1]);
2342.
2343.         object.len = Math.sqrt(Math.pow((object.endY - object.y), 2) + Math.pow((object.endX -
      object.x), 2));
2344.
2345.         if (setupType == 1) {
2346.             object.visible = true;
2347.         }
2348.         else {
2349.             object.visible = false;
2350.         }
2351.
2352.
2353.         object.scaleX = object.len;
2354.         object.scaleY = 1;
2355.
2356.         object.rotation = Math.atan2((object.endY - object.y), (object.endX - object.x)) * 180 /
      Math.PI;
2357.     }
2358.     else if (object.gen != generation) {
2359.         object.alpha -= 3/30;
2360.     }
2361.
2362.     if (object.alpha <= 0) {
2363.         object.alpha = 0;
2364.         object.visible = false;
2365.         object.removeEventListener('tick', setPoses11);
2366.         stage.removeChild(object);
2367.     }
2368.
2369. }
2370.
2371. function setPoses12(e) {
2372.     var object = e.currentTarget;
2373.     if (object.gen == generation) {
2374.         object.x = transfX(yVector[object.num]);
2375.         object.y = transfY(etalon[s3_cur][object.num][s2_cur]);
2376.         object.endX = transfX(yVector[object.num+1]);
2377.         object.endY = transfY(etalon[s3_cur][object.num+1][s2_cur]);
2378.
2379.         object.len = Math.sqrt(Math.pow((object.endY - object.y), 2) + Math.pow((object.endX -
      object.x), 2));
2380.
2381.         if (setupType == 2) {
```

```
            object.visible = true;
        }
        else {
            object.visible = false;
        }


        object.scaleX = object.len;
        object.scaleY = 1;

        object.rotation = Math.atan2((object.endY - object.y), (object.endX - object.x)) * 180 /
    Math.PI;
    }
    else if (object.gen != generation) {
        object.alpha -= 3/30;
    }

    if (object.alpha <= 0) {
        object.alpha = 0;
        object.visible = false;
        object.removeEventListener('tick', setPoses12);
        stage.removeChild(object);
    }

}
function setPoses13(e) {
    var object = e.currentTarget;
    if (object.gen == generation) {
        object.x = transfX(tVector[object.num]);
        object.y = transfY(etalon[object.num][s1_cur][s2_cur]);
        object.endX = transfX(tVector[object.num+1]);
        object.endY = transfY(etalon[object.num+1][s1_cur][s2_cur]);

        object.len = Math.sqrt(Math.pow((object.endY - object.y), 2) + Math.pow((object.endX -
    object.x), 2));

        if (setupType == 3) {
            object.visible = true;
        }
        else {
            object.visible = false;
        }


        object.scaleX = object.len;
        object.scaleY = 1;

        object.rotation = Math.atan2((object.endY - object.y), (object.endX - object.x)) * 180 /
    Math.PI;
    }
    else if (object.gen != generation) {
        object.alpha -= 3/30;
    }

    if (object.alpha <= 0) {
        object.alpha = 0;
        object.visible = false;
        object.removeEventListener('tick', setPoses13);
        stage.removeChild(object);
    }

}



function setPoses21(e) {
    var object = e.currentTarget;
```

```
    if (object.gen == generation) {
        object.x = transfX(xVector[object.num]);
        object.y = transfY(solve[s3_cur][s1_cur][object.num]);
        object.endX = transfX(xVector[object.num+1]);
        object.endY = transfY(solve[s3_cur][s1_cur][object.num+1]);

        object.len = Math.sqrt(Math.pow((object.endY - object.y), 2) + Math.pow((object.endX - object.x), 2));

        if (setupType == 1) {
            object.visible = true;
        }
        else {
            object.visible = false;
        }


        object.scaleX = object.len;
        object.scaleY = 1;

        object.rotation = Math.atan2((object.endY - object.y), (object.endX - object.x)) * 180 / Math.PI;
    }
    else if (object.gen != generation) {
        object.alpha -= 3/30;
    }

    if (object.alpha <= 0) {
        object.alpha = 0;
        object.visible = false;
        object.removeEventListener('tick', setPoses21);
        stage.removeChild(object);
    }

}

function setPoses22(e) {
    var object = e.currentTarget;
    if (object.gen == generation) {
        object.x = transfX(yVector[object.num]);
        object.y = transfY(solve[s3_cur][object.num][s2_cur]);
        object.endX = transfX(yVector[object.num+1]);
        object.endY = transfY(solve[s3_cur][object.num+1][s2_cur]);

        object.len = Math.sqrt(Math.pow((object.endY - object.y), 2) + Math.pow((object.endX - object.x), 2));

        if (setupType == 2) {
            object.visible = true;
        }
        else {
            object.visible = false;
        }


        object.scaleX = object.len;
        object.scaleY = 1;

        object.rotation = Math.atan2((object.endY - object.y), (object.endX - object.x)) * 180 / Math.PI;
    }
    else if (object.gen != generation) {
        object.alpha -= 3/30;
    }

    if (object.alpha <= 0) {
        object.alpha = 0;
        object.visible = false;
```

```
        object.removeEventListener('tick', setPoses22);
        stage.removeChild(object);
    }

}

function setPoses23(e) {
    var object = e.currentTarget;
    if (object.gen == generation) {
        object.x = transfX(tVector[object.num]);
        object.y = transfY(solve[object.num][s1_cur][s2_cur]);
        object.endX = transfX(tVector[object.num+1]);
        object.endY = transfY(solve[object.num+1][s1_cur][s2_cur]);

        object.len = Math.sqrt(Math.pow((object.endY - object.y), 2) + Math.pow((object.endX - object.x), 2));

        if (setupType == 3) {
            object.visible = true;
        }
        else {
            object.visible = false;
        }


        object.scaleX = object.len;
        object.scaleY = 1;

        object.rotation = Math.atan2((object.endY - object.y), (object.endX - object.x)) * 180 / Math.PI;
    }
    else if (object.gen != generation) {
        object.alpha -= 3/30;
    }

    if (object.alpha <= 0) {
        object.alpha = 0;
        object.visible = false;
        object.removeEventListener('tick', setPoses23);
        stage.removeChild(object);
    }

}


function setPoses31(e) {
    var object = e.currentTarget;
    if (object.gen == generation) {
        object.x = transfEX(xVector[object.num]);
        object.y = transfEY(errorCube[s3_cur][s1_cur][object.num]);
        object.endX = transfEX(xVector[object.num+1]);
        object.endY = transfEY(errorCube[s3_cur][s1_cur][object.num+1]);

        object.len = Math.sqrt(Math.pow((object.endY - object.y), 2) + Math.pow((object.endX - object.x), 2));

        if (setupType == 1 && pogr_type == 1) {
            object.visible = true;
        }
        else {
            object.visible = false;
        }


        object.scaleX = object.len;
        object.scaleY = 1;

```

```
        object.rotation = Math.atan2((object.endY - object.y), (object.endX - object.x)) * 180 / Math.PI;
    }
    else if (object.gen != generation) {
        object.alpha -= 3/30;
    }

    if (object.alpha <= 0) {
        object.alpha = 0;
        object.visible = false;
        object.removeEventListener('tick', setPoses21);
        stage.removeChild(object);
    }

}

function setPoses32(e) {
    var object = e.currentTarget;
    if (object.gen == generation) {
        object.x = transfEX(yVector[object.num]);
        object.y = transfEY(errorCube[s3_cur][object.num][s2_cur]);
        object.endX = transfEX(yVector[object.num+1]);
        object.endY = transfEY(errorCube[s3_cur][object.num+1][s2_cur]);

        object.len = Math.sqrt(Math.pow((object.endY - object.y), 2) + Math.pow((object.endX - object.x), 2));

        if (setupType == 2 && pogr_type == 1) {
            object.visible = true;
        }
        else {
            object.visible = false;
        }


        object.scaleX = object.len;
        object.scaleY = 1;

        object.rotation = Math.atan2((object.endY - object.y), (object.endX - object.x)) * 180 / Math.PI;
    }
    else if (object.gen != generation) {
        object.alpha -= 3/30;
    }

    if (object.alpha <= 0) {
        object.alpha = 0;
        object.visible = false;
        object.removeEventListener('tick', setPoses22);
        stage.removeChild(object);
    }

}

function setPoses33(e) {
    var object = e.currentTarget;
    if (object.gen == generation) {
        object.x = transfEX(tVector[object.num]);
        object.y = transfEY(errorCube[object.num][s1_cur][s2_cur]);
        object.endX = transfEX(tVector[object.num+1]);
        object.endY = transfEY(errorCube[object.num+1][s1_cur][s2_cur]);

        object.len = Math.sqrt(Math.pow((object.endY - object.y), 2) + Math.pow((object.endX - object.x), 2));

        if (setupType == 3 && pogr_type == 1) {
            object.visible = true;
        }
```

```
        else {
            object.visible = false;
        }


        object.scaleX = object.len;
        object.scaleY = 1;

        object.rotation = Math.atan2((object.endY - object.y), (object.endX - object.x)) * 180 /
    Math.PI;
    }
    else if (object.gen != generation) {
        object.alpha -= 3/30;
    }

    if (object.alpha <= 0) {
        object.alpha = 0;
        object.visible = false;
        object.removeEventListener('tick', setPoses23);
        stage.removeChild(object);
    }

}


function setPoses41(e) {
    var object = e.currentTarget;
    if (object.gen == generation) {
        object.x = transfE2X(xVector[object.num]);
        object.endX = transfE2X(xVector[object.num+1]);

        if (setupType == 2 && pogr_type == 3) {
            object.y = transfEY(EotTx[object.num]);
            object.endY = transfEY(EotTx[object.num+1]);
        }
        else {
            object.y = transfEY(EotTx2[object.num]);
            object.endY = transfEY(EotTx2[object.num+1]);
        }

        object.len = Math.sqrt(Math.pow((object.endY - object.y), 2) + Math.pow((object.endX -
    object.x), 2));

        if ((setupType == 2 && pogr_type == 3) || (setupType == 3 && pogr_type == 2)) {
            object.visible = true;
        }
        else {
            object.visible = false;
        }


        object.scaleX = object.len;
        object.scaleY = 1;

        object.rotation = Math.atan2((object.endY - object.y), (object.endX - object.x)) * 180 /
    Math.PI;
    }
    else if (object.gen != generation) {
        object.alpha -= 3/30;
    }

    if (object.alpha <= 0) {
        object.alpha = 0;
        object.visible = false;
        object.removeEventListener('tick', setPoses41);
        stage.removeChild(object);
    }

```

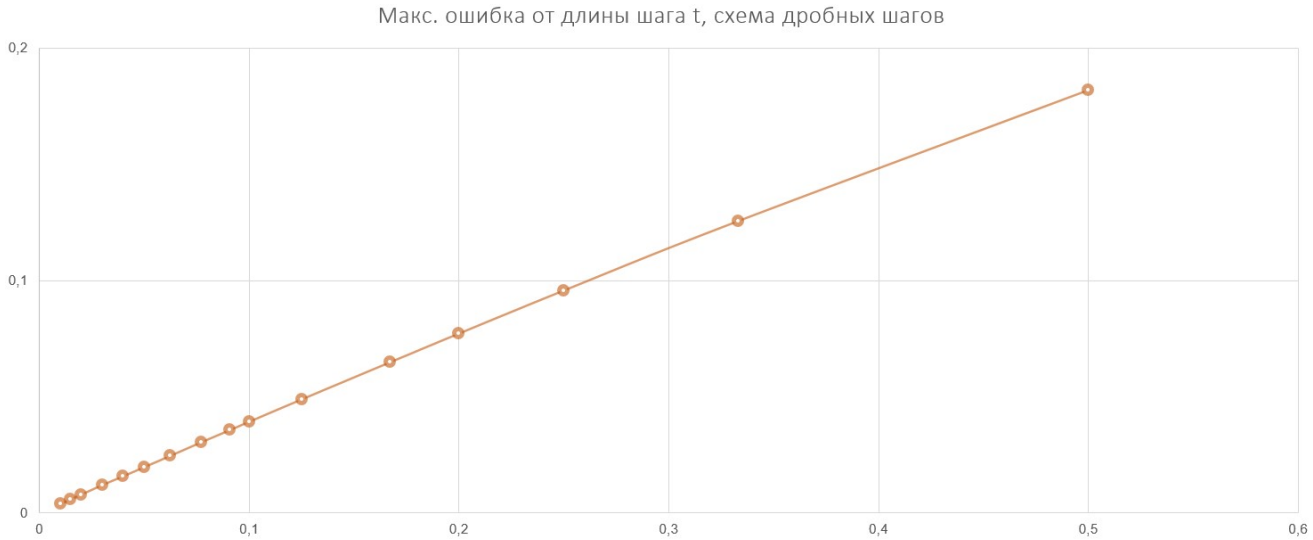
```
}


function setPoses42(e) {
    var object = e.currentTarget;
    if (object.gen == generation) {
        object.x = transfE2X(yVector[object.num]);
        object.endX = transfE2X(yVector[object.num+1]);

        if (setupType == 1 && pogr_type == 3) {
            object.y = transfEY(EotTy[object.num]);
            object.endY = transfEY(EotTy[object.num+1]);
        }
        else {
            object.y = transfEY(EotTy2[object.num]);
            object.endY = transfEY(EotTy2[object.num+1]);
        }

        object.len = Math.sqrt(Math.pow((object.endY - object.y), 2) + Math.pow((object.endX - object.x), 2));

        if ((setupType == 1 && pogr_type == 3) || (setupType == 3 && pogr_type == 3)) {
            object.visible = true;
        }
        else {
            object.visible = false;
        }


        object.scaleX = object.len;
        object.scaleY = 1;

        object.rotation = Math.atan2((object.endY - object.y), (object.endX - object.x)) * 180 / Math.PI;
    }
    else if (object.gen != generation) {
        object.alpha -= 3/30;
    }

    if (object.alpha <= 0) {
        object.alpha = 0;
        object.visible = false;
        object.removeEventListener('tick', setPoses42);
        stage.removeChild(object);
    }

}


function setPoses43(e) {
    var object = e.currentTarget;
    if (object.gen == generation) {
        object.x = transfE2X(tVector[object.num]);
        object.endX = transfE2X(tVector[object.num+1]);

        if (setupType == 1 && pogr_type == 2) {
            object.y = transfEY(EotTt[object.num]);
            object.endY = transfEY(EotTt[object.num+1]);
        }
        else {
            object.y = transfEY(EotTt2[object.num]);
            object.endY = transfEY(EotTt2[object.num+1]);
        }

        object.len = Math.sqrt(Math.pow((object.endY - object.y), 2) + Math.pow((object.endX - object.x), 2));

        if ((setupType == 1 && pogr_type == 2) || (setupType == 2 && pogr_type == 2)) {
```

```
            object.visible = true;
        }
        else {
            object.visible = false;
        }


        object.scaleX = object.len;
        object.scaleY = 1;

        object.rotation = Math.atan2((object.endY - object.y), (object.endX - object.x)) * 180 /
    Math.PI;
    }
    else if (object.gen != generation) {
        object.alpha -= 3/30;
    }

    if (object.alpha <= 0) {
        object.alpha = 0;
        object.visible = false;
        object.removeEventListener('tick', setPoses43);
        stage.removeChild(object);
    }

}




//метод прогонки
function progonka(matrix, vectorB) {
    var vectorX = [];

    var N = vectorB.length;

    var alphas = [];
    var betas = [];


    for (var i = 0 ; i < vectorB.length ; ++i) {
        alphas.push(0);
        betas.push(0);
    }

    //Прямой ход прогонки

    for (var i = 0; i < N; ++i) {
        var A0, C0, B0, F0;

        if (i - 1 < 0) {
            A0 = 0;
        }
        else {
            A0 = matrix[i][i - 1];
        }

        C0 = -1 * matrix[i][i];

        if (i + 1 < N) {
            B0 = matrix[i][i + 1];
        }
        else {
            B0 = 0;
        }

        F0 = vectorB[i];
```

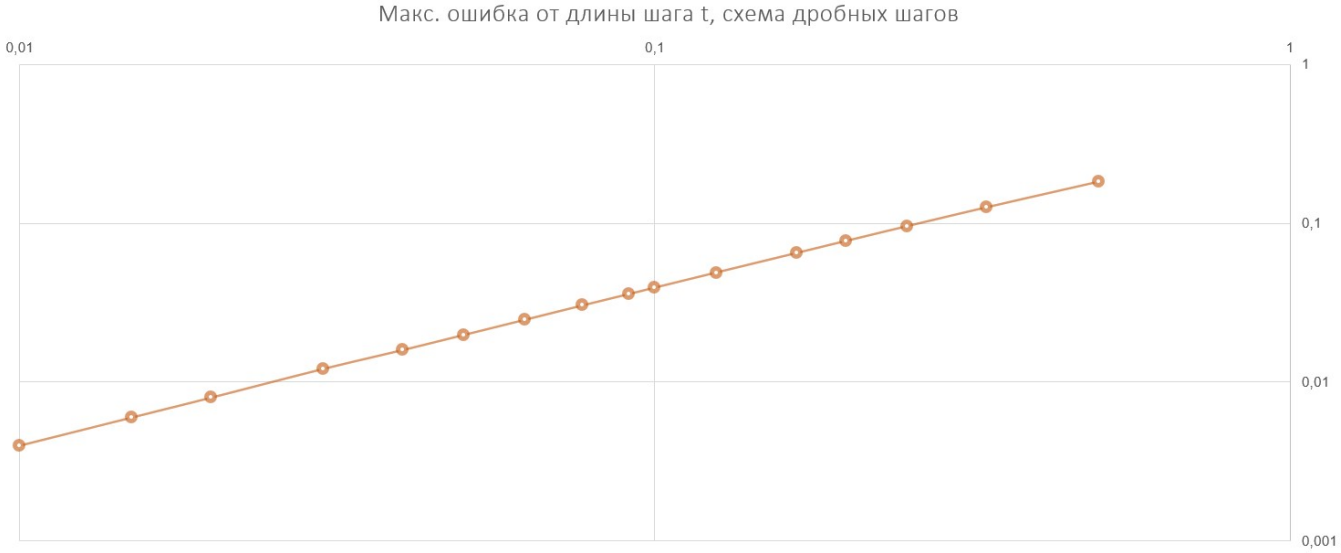
```
2836.
2837.
2838.         if (i == 0) {
2839.             alphas[i] = B0 / C0;
2840.             betas[i] = -(F0 / C0);
2841.         }
2842.         else if (i == N - 1) {
2843.             alphas[i] = 0;
2844.             betas[i] = (betas[i - 1] * A0 - F0) / (C0 - alphas[i - 1] * A0);
2845.         }
2846.         else {
2847.             alphas[i] = B0 / (C0 - alphas[i - 1] * A0);
2848.             betas[i] = (betas[i - 1] * A0 - F0) / (C0 - alphas[i - 1] * A0);
2849.         }
2850.
2851.     }
2852.
2853.
2854.     vectorX[N - 1] = betas[N - 1];
2855.
2856.     for (var i = 2; i <= N; ++i) {
2857.         vectorX[N - i] = alphas[N - i] * vectorX[N - i + 1] + betas[N - i];
2858.     }
2859.
2860.     return vectorX;
2861. }
```

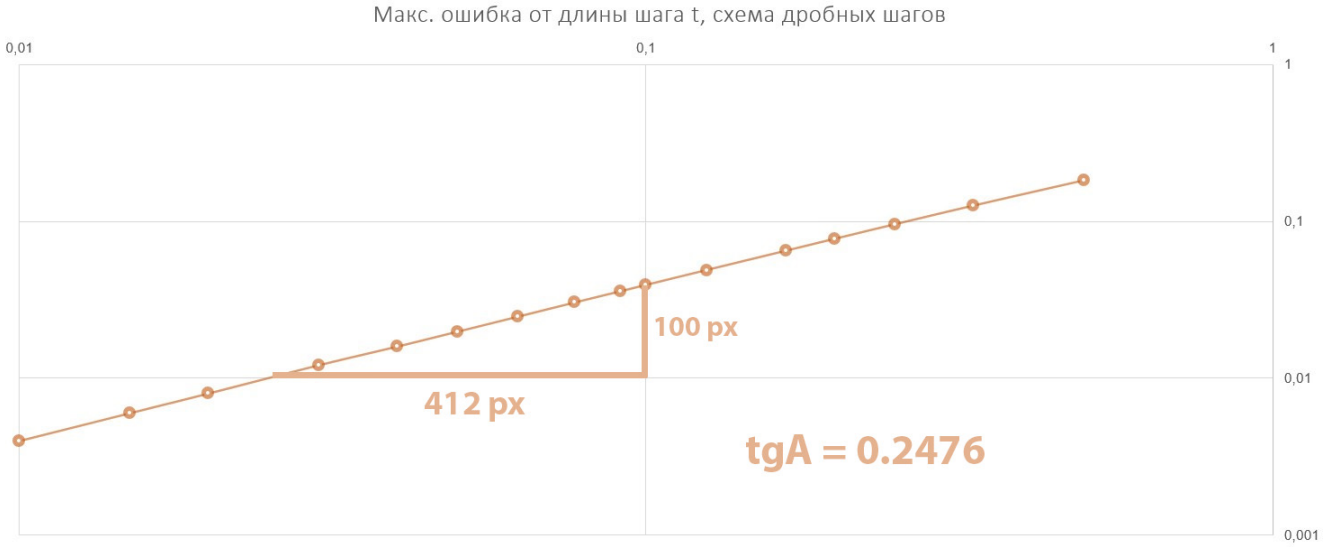
7. Графики

При T = 18



Видим, что ошибка растет линейно с размером шага.





Данная лабораторная работа выполнена: **22 января 2021.**

Вывод по лабораторным

Выполняя данные лабораторные, я познакомился с различными конечно-разностными схемами. Они позволяют решить многие задачи математической физики, которые решать аналитически или напрямую будет очень сложно. Однако, результаты не всегда бывают точными, существует хоть какая-то, но погрешность. В зависимости от того, как растет погрешность с шагом, можно определить какой порядок аппроксимации уравнения. На самом деле, было очень сложно больше сделать весь рабочий интерфейс, запрограммировать отрисовку, выбор параметров, динамические графики и графики погрешностей. Все это не сильно связано с самим предметом, однако, это улучшило мои основные навыки программирования и дизайна, что для меня очень важно. Не знаю, придется ли мне еще столкнуться с похожими задачами, но было интересно решать их с помощью схем. Самой лютой и сложной лабораторной оказалась последняя. Примерно 12 часов непрерывной работы (пишу этот отчет в 8 утра и до сих пор даже не спал) и почти 3 тысячи строк кода ушло на то, чтобы запрограммировать ее, сделать рабочий интерфейс, графики, однако, даже сейчас присутствуют небольшие косяки. Пришлось столкнуться с большой проблемой, когда схема в последней лабораторной не работала как нужно. Дело в том, что я с помощью нее только решал середину (методом прогонки), а нужно было весь участок, включая границы. Можно много еще чего написать в выводе, однако, думаю, что эта работа была довольно полезной для меня, хотя бы я усвоил свои навыки и узнал что-то новое.