Eigenstructure of the Butterfly Scheme

Denis Zorin, February 1998

This worksheet contains the symbolic part of the analysis of tangen plane continuity and C1-continuity of the Butterfly scheme. We compute the eigenvalues of the subdivision matrix with the maximal magnitude and the corresponding eigenvectors. For the valence K = 3, the largest eigenvalue is 1/4, with 3 Jordan blocks: two of size 2 and one of size 3. For K = 4,5,7 the largest eigenvalues are in the first and last blocks of the DFT-transformed matrix, and have trivial Jordan blocks. For K >= 8, the largest eigenvalues are in other blocks. We generate the C-code for the computationally intensive part of the analysis (analysis of the characteristic maps). The generated code uses functions from our wrapper class for f.p. numbers, encapsulating interval arithmetics.

Utilities

Subdivision matrix

```
> assume( c >= -1); additionally( c <= -1); additionally (c, real); assume( K, integer);
  additionally( K >= 3);
> Butterfly := matrix( [
   [ (1/2) + 4*w*c - 2*w*(2*c^2-1), 0,- w*(conjugate(omega) + 1),0,0,0],
   [1,0,0,0,0,0],
   [(1/2)*(1+ omega) - w*(conjugate(omega) + omega^2 ), -w*(1 + omega),2*w,0,0,0],
   [1/2 - 2*w*c,1/2,2*w*(1+conjugate(omega)),0, -w, -w*conjugate(omega)],
   [1/2 + 2*w*omega, 2*w - w*omega,1/2-w*conjugate(omega),0,-w,0],
   [(1/2)*omega + 2*w,2*w*omega-w, 1/2 - w*omega,0,0,-w]]);
```

$$Butterfly := \begin{bmatrix} \frac{1}{2}+4\,w\,c\sim-2\,w\,(2\,c\sim^2-1) & 0 & -w\,(\overline{\omega}+1) & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{2}+\frac{1}{2}\omega-w\,(\overline{\omega}+\omega^2) & -w\,(1+\omega) & 2\,w & 0 & 0 & 0 \\ \frac{1}{2}-2\,w\,c\sim & \frac{1}{2} & 2\,w\,(\overline{\omega}+1) & 0 & -w & -w\,\overline{\omega} \\ \frac{1}{2}+2\,w\,\omega & 2\,w-w\,\omega & \frac{1}{2}-w\,\overline{\omega} & 0 & -w & 0 \\ \frac{1}{2}\omega+2\,w & 2\,w\,\omega-w & \frac{1}{2}-w\,\omega & 0 & 0 & -w \end{bmatrix}$$

```
> Butterconst := { w = 1/16}; Buttervar := { omega = exp(2*I*Pi*m/K), c = cos(2*m*Pi/K) };
```

$$Butterconst := \{ w = \frac{1}{16} \}$$

$$Buttervar := \{ c\sim = \cos\left(2\,\frac{m\,\pi}{K\sim}\right), \omega = \mathbf{e}^{\left(2\frac{I\,\pi\,m}{K\sim}\right)} \}$$

```
> ButterflyExpanded :=  map( simplify, subs( s^2 = 1-c^2, map( simplify, map( evalc, subs( { omega = c + s*I , op(Butterconst)},
  eval(Butterfly))) )))
  ;
```

$$ButterflyExpanded := \begin{bmatrix} \frac{5}{8}+\frac{1}{4}c\sim-\frac{1}{4}c\sim^2 & 0 & -\frac{1}{16}c\sim-\frac{1}{16}+\frac{1}{16}I\,s & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ \frac{9}{16}+\frac{7}{16}c\sim-\frac{1}{8}c\sim^2+\frac{9}{16}I\,s-\frac{1}{8}I\,c\sim\,s & -\frac{1}{16}-\frac{1}{16}c\sim-\frac{1}{16}I\,s & \frac{1}{8} & 0 & 0 & 0 \\ \frac{1}{2}-\frac{1}{8}c\sim & \frac{1}{2} & \frac{1}{8}c\sim+\frac{1}{8}-\frac{1}{8}I\,s & 0 & \frac{-1}{16} & -\frac{1}{16}c\sim+\frac{1}{16}I\,s \\ \frac{1}{2}+\frac{1}{8}c\sim+\frac{1}{8}I\,s & \frac{1}{8}-\frac{1}{16}c\sim-\frac{1}{16}I\,s & \frac{1}{2}-\frac{1}{16}c\sim+\frac{1}{16}I\,s & 0 & \frac{-1}{16} & 0 \\ \frac{1}{2}c\sim+\frac{1}{2}I\,s+\frac{1}{8} & \frac{1}{8}c\sim+\frac{1}{8}I\,s-\frac{1}{16} & \frac{1}{2}-\frac{1}{16}c\sim-\frac{1}{16}I\,s & 0 & 0 & \frac{-1}{16} \end{bmatrix}$$

Because the matrix has block-diagonal structure, we have to compute only the eigenvalues of the subblocks on the diagonal.

```
> Butterfly00 := submatrix( ButterflyExpanded, 1..3,1..3):
> Butterfly11 := submatrix( ButterflyExpanded, 4..6,4..6):
> Butterfly10 := submatrix( ButterflyExpanded, 4..6,1..3):
>
```

Introduce new variables, cs and ss, for $\cos\left(\frac{m\,\pi}{K}\right)$ and $\sin\left(\frac{m\,\pi}{K}\right)$.

```
> Cre :=  diag( 1, 1, cs - I*ss );
```

$$Cre := \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & cs - I\,ss \end{bmatrix}$$

Reduce the first subblock to a real matrix using a coordinate transoform. Eigenvalues do not change.

```
> Butter00re :=  map( simplify, subs( ss^2 = 1- cs^2, map( expand, map( simplify, subs( {s^2 = 1 - c^2}, subs( { c = 2*cs^2 - 1, s =
  2*cs*ss},  map(simplify, map( evalc, map( simplify, evalm( Cre &*  eval(Butterfly00) &* inverse(Cre)))))))))));
```

$$Butter00re := \begin{bmatrix} \frac{1}{8}+\frac{3}{2}cs^2-cs^4 & 0 & -\frac{1}{8}cs \\ 1 & 0 & 0 \\ -\frac{1}{2}cs^3+\frac{11}{8}cs & -\frac{1}{8}cs & \frac{1}{8} \end{bmatrix}$$

Analysis of the behavior of the eigenvalues

Our goal is to determine the expresions for the eigenvalues of the largest magnitude, excluding 1, and show that for valences > 8 these eigenvalues are not in the 1st and last blocks of the DFT-transformed subdivision matrix. With some additional easily provable assumptions, this means that the Butterfly scheme is not C1 for these valences. We also explicitly compute the eigenvalues for K = 3.

0-th block

The block corresponding to m = 0 is present in every matrix; if the eigenvalues of some other block are greater than 1/4, this block is not dominant.

```
> jordan(subs(  cs = 1, eval(Butter00re)));
```

$$\begin{bmatrix} \frac{1}{4} & 1 & 0 \\ 0 & \frac{1}{4} & 1 \\ 0 & 0 & \frac{1}{4} \end{bmatrix}$$

Characteristic polynomial of the first sublock and its descriminant

The eigenvalues of the second subblock are 0, and -1/16; we will see that the first subblock always has larger eigenvalues.

Characteristic polynomial

```
> ButterCharpoly := subs( cs = sqrt(d), collect( subs( cos(m*Pi/K) = c, expand( charpoly(Butter00re,lambda),trig)), lambda));
```

$$ButterCharpoly := \lambda^3 + \left(-\frac{1}{4} + d^2 - \frac{3}{2}d\right)\lambda^2 + \left(\frac{23}{64}d + \frac{1}{64} - \frac{3}{16}d^2\right)\lambda - \frac{1}{64}d$$

```
> re := coeff( ButterCharpoly, lambda^2); se := coeff( ButterCharpoly, lambda); te := rem(ButterCharpoly, lambda, lambda);
```

$$re := -\frac{1}{4} + d^2 - \frac{3}{2}d$$

$$se := \frac{23}{64}d + \frac{1}{64} - \frac{3}{16}d^2$$

$$te := -\frac{1}{64}d$$

Reduce the characteristic polynomial; use $d = c^2$ as the parameter.

```
> ButterCharpolyReduced := collect(simplify(subs( lambda = mu - re/3, ButterCharpoly)), mu);
```

$$ButterCharpolyReduced := \mu^3 + \left(-\frac{1}{192} + d^3 - \frac{37}{48}d^2 + \frac{7}{64}d - \frac{1}{3}d^4\right)\mu + \frac{1}{768}d + \frac{55}{1152}d^2 + \frac{73}{144}d^4 - \frac{19}{64}d^3 + \frac{2}{27}d^6 - \frac{1}{3}d^5 + \frac{1}{6912}$$

```
> pe := coeff(ButterCharpolyReduced, mu); qe := rem(ButterCharpolyReduced, mu,mu);
```

$$pe := -\frac{1}{192} + d^3 - \frac{37}{48}d^2 + \frac{7}{64}d - \frac{1}{3}d^4$$

$$qe := \frac{1}{768}d + \frac{55}{1152}d^2 + \frac{73}{144}d^4 - \frac{19}{64}d^3 + \frac{2}{27}d^6 - \frac{1}{3}d^5 + \frac{1}{6912}$$

Find the discriminant and deterimine its sign.

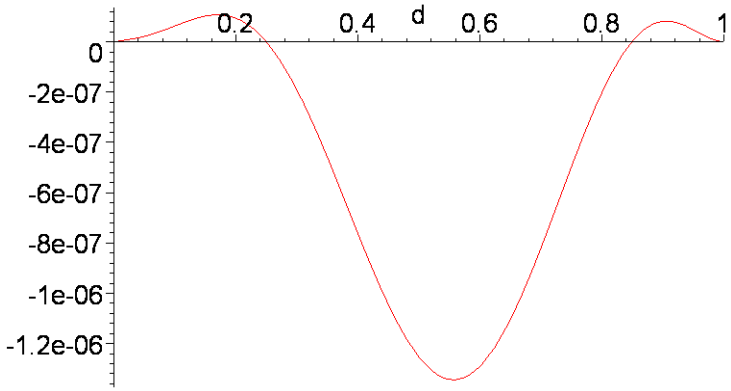
```
> Discr := simplify( (pe/3)^3 + (qe/2)^2);
```

$$Discr := \frac{1}{2359296}d - \frac{19}{3538944}d^2 - \frac{479}{442368}d^4 + \frac{1123}{7077888}d^3 - \frac{1369}{442368}d^6 + \frac{299}{110592}d^5 + \frac{91}{55296}d^7 - \frac{1}{3072}d^8$$

```
> Discr := factor(subs( w = 1/16, Discr));
```

$$Discr := -\frac{1}{7077888}d\,(4d-1)\,(576d^4 - 1616d^3 + 976d^2 - 20d + 3)\,(d-1)^2$$

```
> plot(Discr, d = 0..1);
```

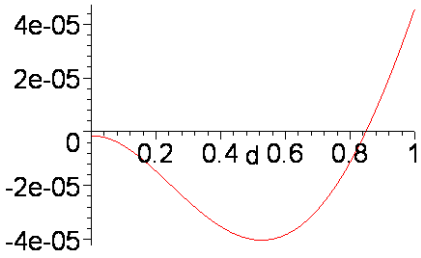


Pull out the degree 4 factor responcible for one of the roots on 0..1

```
DiscrFactor4 := factor(Discr/(lcoeff(Discrs)*(d-1/4)*(d-1)^2*d));
```

$$DiscrFactor4 := -\frac{1}{3072}d^4 + \frac{101}{110592}d^3 - \frac{61}{110592}d^2 + \frac{5}{442368}d - \frac{1}{589824}$$

```
> plot(DiscrFactor4, d=0..1);
```



Find the interesting root

```
> DiscrRoot := solve( { DiscrFactor4 = 0, d <= 1, d >= 0},d );
```

$$DiscrRoot := \left\{ d = \frac{101}{144} + \frac{1}{144}\sqrt{\frac{4345\,(13061314+5238\sqrt{229017})^{1/3} + 12\,(13061314+5238\sqrt{229017})^{2/3} + 657264}{(13061314+5238\sqrt{229017})^{1/3}}} - \frac{1}{144}\left(\left(\right.\right.\right.$$
$$8690\,(13061314+5238\sqrt{229017})^{1/3}\sqrt{\frac{4345\,(13061314+5238\sqrt{229017})^{1/3} + 12\,(13061314+5238\sqrt{229017})^{2/3} + 657264}{(13061314+5238\sqrt{229017})^{1/3}}}$$
$$- 12\sqrt{\frac{4345\,(13061314+5238\sqrt{229017})^{1/3} + 12\,(13061314+5238\sqrt{229017})^{2/3} + 657264}{(13061314+5238\sqrt{229017})^{1/3}}}\,(13061314+5238\sqrt{229017})^{2/3}$$
$$- 657264\sqrt{\frac{4345\,(13061314+5238\sqrt{229017})^{1/3} + 12\,(13061314+5238\sqrt{229017})^{2/3} + 657264}{(13061314+5238\sqrt{229017})^{1/3}}} + 312154\,(13061314+5238\sqrt{229017})^{1/3}\Bigg) \Big/ \Bigg($$
$$\left.\left.\left.(13061314+5238\sqrt{229017})^{1/3}\sqrt{\frac{4345\,(13061314+5238\sqrt{229017})^{1/3} + 12\,(13061314+5238\sqrt{229017})^{2/3} + 657264}{(13061314+5238\sqrt{229017})^{1/3}}}\right)\right)^{1/2}\right\}$$

```
> DiscrRoot := op(2,op(DiscrRoot));
```

$$DiscrRoot := \frac{101}{144} + \frac{1}{144}\sqrt{\frac{4345\,(13061314+5238\sqrt{229017})^{1/3} + 12\,(13061314+5238\sqrt{229017})^{2/3} + 657264}{(13061314+5238\sqrt{229017})^{1/3}}} - \frac{1}{144}\left(\left(\right.\right.\right.$$
$$8690\,(13061314+5238\sqrt{229017})^{1/3}\sqrt{\frac{4345\,(13061314+5238\sqrt{229017})^{1/3} + 12\,(13061314+5238\sqrt{229017})^{2/3} + 657264}{(13061314+5238\sqrt{229017})^{1/3}}}$$
$$- 12\sqrt{\frac{4345\,(13061314+5238\sqrt{229017})^{1/3} + 12\,(13061314+5238\sqrt{229017})^{2/3} + 657264}{(13061314+5238\sqrt{229017})^{1/3}}}\,(13061314+5238\sqrt{229017})^{2/3}$$
$$- 657264\sqrt{\frac{4345\,(13061314+5238\sqrt{229017})^{1/3} + 12\,(13061314+5238\sqrt{229017})^{2/3} + 657264}{(13061314+5238\sqrt{229017})^{1/3}}} + 312154\,(13061314+5238\sqrt{229017})^{1/3}\Bigg) \Big/ \Bigg($$
$$\left.\left.\left.(13061314+5238\sqrt{229017})^{1/3}\sqrt{\frac{4345\,(13061314+5238\sqrt{229017})^{1/3} + 12\,(13061314+5238\sqrt{229017})^{2/3} + 657264}{(13061314+5238\sqrt{229017})^{1/3}}}\right)\right)^{1/2}$$

```
> evalf(DiscrRoot);
```

.8486812039

We observe that the discriminant has four roots: 0,1, 1/4 and approx. 0.8486812039; these are the only values for which the matrix may have nontrivial Jordan blocks. The last case does not occur in the cases which are of interest to us. In the first 3 cases the Jordan normal form can be find explicitly.

The case of three real roots

The discriminant is positive on 0..1/4 and on DiscrRoot..1, negative on 1/4..DiscrRoot

Compute the solutions when the discriminant is negative and, therefore, there are 3 real roots

```
> R :=  sqrt(-factor(pe)/3):
> phi := arccos( qe/(2*R^3)):
> r1 := -2*R*cos(phi/3)-re/3: r2 :=  -2*R*cos(phi/3 + 2*Pi/3)-re/3: r3 :=  -2*R*cos(phi/3 + 4*Pi/3)-re/3:
```

The product of the roots is -d/64, which is negative; therefore, either all three are positive, or two are negative. 0 is never a root, except when d = 0

The roots cannot be equal on the interval where the discriminant does not change sign; we can figure out the largest one on the whole interval by evaluating them at a single value of d. We conclude that all three roots are always positive and the largest one is the second.

d = 1/4..DiscrRoot. We use interval arithmetics to guarantee correctness.

```
> InvervalRoots := map( unapply('inapply'(x,d),x), [r1,r2,r3] ):
> eval(InvervalRoots(1/2));
```

[[.08499852729, .08499853961], [.4690415185, .4690415259], [.1959599360, .1959599530]]

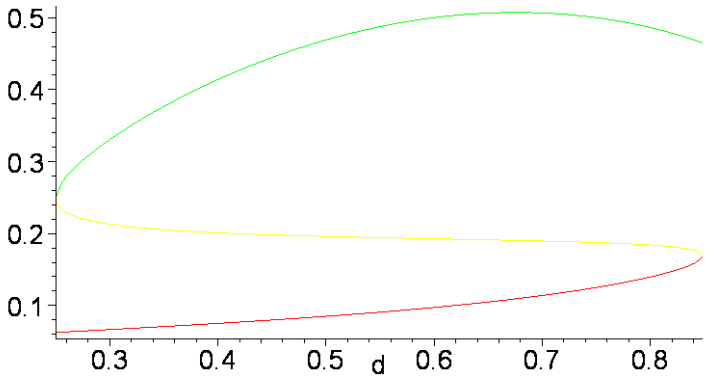
We conclude that te second root is the largest.

```
> AbsDominantEV1 := r2;
```

$$AbsDominantEV1 := \frac{1}{12}\sqrt{(d-1)\,(64\,d^3 - 128\,d^2 + 20\,d - 1)}\,\sin\left(\frac{1}{3}\arccos\left(6912\,\frac{\frac{1}{768}d + \frac{55}{1152}d^2 + \frac{73}{144}d^4 - \frac{19}{64}d^3 + \frac{2}{27}d^6 - \frac{1}{3}d^5 + \frac{1}{6912}}{((d-1)\,(64\,d^3 - 128\,d^2 + 20\,d - 1))^{3/2}}\right) + \frac{1}{6}\pi\right) + \frac{1}{12} - \frac{1}{3}d^2 + \frac{1}{2}d$$

[[.08499852699, .08499853971], [.4690415184, .4690415260], [.1959599356, .1959599533]]

```
> plot( [r1, r2, r3], d = 1/4..DiscrRoot);
```



The case of one real root

First case: The interval 0..1/4

We show that for d in this interval, all roots are < 1/4; we will see that there is always an eigenvalue > 1/4 elsewhere. Therefore, the roots on this interval are irrelevant. There is only one real root; if at a point x the value of the polynomial is positive, than the magnitude of the real root is less than x. We see that the char. polynomial is positive at 1/4 for d = 0..1/4, and negative for d = 1..1/4. For any K > 3, $\frac{1}{4} < \cos\left(\frac{\pi}{K}\right)^2$, therfore, there is a real eigenvalue of magnitude greater than 1/4.

```
> solve( subs( lambda = 1/4, ButterCharpoly) > 0 );
```

$$\text{RealRange}\left(-\infty, \text{Open}\left(\frac{1}{4}\right)\right), \text{RealRange}(\text{Open}(1), \infty)$$

Now build an equation for the square of the magnitude of the other two roots; it is a cubic equation again:

```
> rsq := - se; ssq := expand(te*re); tsq := -te*te;
```

$$rsq := -\frac{23}{64}d - \frac{1}{64} + \frac{3}{16}d^2$$

$$ssq := \frac{1}{256}d - \frac{1}{64}d^3 + \frac{3}{128}d^2$$

$$tsq := -\frac{1}{4096}d^2$$

Use the same approach: verifying that all solutions are < 1/16 on d=1..1/4

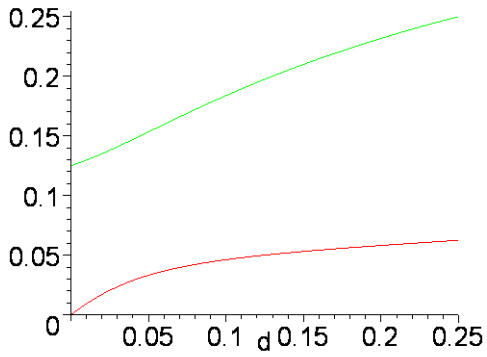
```
> solve( subs(x = 1/16, x^3 + rsq*x^2 + ssq*x + tsq) > 0);
```

$$\text{RealRange}\left(-\infty, \text{Open}\left(\frac{1}{4}\right)\right), \text{RealRange}\left(\text{Open}\left(\frac{3}{4}\right), \text{Open}(1)\right)$$

For plots, get the expressions for the roots

```
> R :=  signum(qe)*sqrt(-pe/3): phi := arccosh( abs(qe)/(2*abs(R)^3)):
```

```
> r := -2*R*cosh(phi/3)-re/3: c1 := R*( cosh(phi/3) + I*sqrt(3)*sinh(phi/3))-re/3: c2 := R*( cosh(phi/3) +
  I*sqrt(3)*sinh(phi/3)) -re/3:
> plot( [ abs(r), abs(c1), abs(c2)], d=0..1/4);
```



Second case: interval DiscrRoot..1

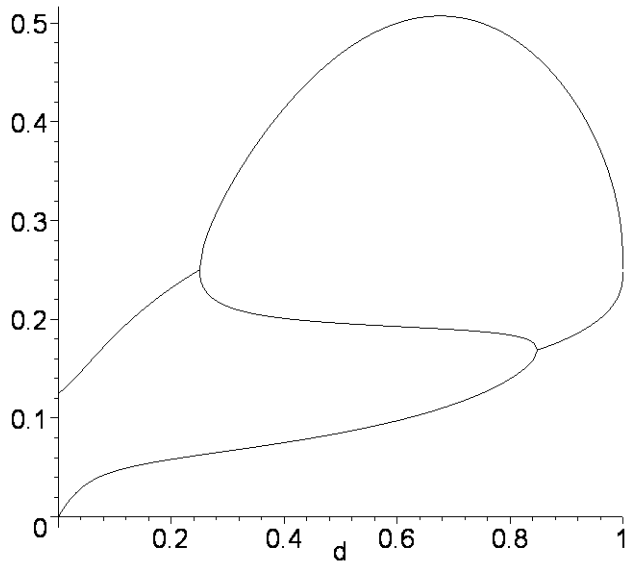
In this case, we show that the real root is the largest; we have already observed that on 3/4 ..1 the magnitude of the complex roots (when there are complex roots) is < 1/4; hence, it is sufficien to show that the real root is greater than 1/4. But we have seen already, that the characteristic polynomial is negative at 1/4 for d > 1/4. Therefore, the real root > 1/4.

```
> R :=  sqrt(-pe/3): phi := arccosh( abs(qe)/(2*abs(R)^3)): AbsDominantEV2 := 2*R*cosh(phi/3)-re/3;
```

$$AbsDominantEV2 := \frac{1}{12}\sqrt{1 - 192\,d^3 + 148\,d^2 - 21\,d + 64\,d^4}\,\cosh\left(\frac{1}{3}\,\text{arccosh}\left(6912\,\frac{\left|\frac{1}{768}d + \frac{55}{1152}d^2 + \frac{73}{144}d^4 - \frac{19}{64}d^3 + \frac{2}{27}d^6 - \frac{1}{3}d^5 + \frac{1}{6912}\right|}{\left|1 - 192\,d^3 + 148\,d^2 - 21\,d + 64\,d^4\right|^{3/2}}\right)\right) + \frac{1}{12} - \frac{1}{3}d^2 + \frac{1}{2}d$$

Plot of all roots

```
> display(plot( [abs(r),abs(c1)], d = 0..1/4,  color=black), plot([r1,r2,r3],d=1/4..DiscrRoot,  color=black),
  plot([abs(r),abs(c1)],d=DiscrRoot..1.000001,  color=black));
```



The magnitude of the subdominant eigenvalue decreases for d from approximately 0.67600423 to 1

To show that the subdominant eigenvalues of the Butterfly scheme are not in the correct block for sufficiently large valences, we show that the largest eigenvalue decreases as a function of near 1. To do this, we find the sign of the derivative; it is suficient to evaluate the derivative at a single point, and to show that the derivative is not zero anywhere on an interval. Let $y(d)$ be the root as a function of d. Then differentiating the equation $y^3 + r(d)\,y^2 + s(d)\,y + t(d) = 0$, and setting $\frac{\partial}{\partial d} y$ to zero, we observe that at a point d_0 where the derivative is zero, the value $y(d_0)$ satisfies the equation $\left(\frac{\partial}{\partial d} r\right) y^2 + \left(\frac{\partial}{\partial d} s\right) y + \left(\frac{\partial}{\partial d} t\right) = 0$. It is sufficient to show that for d on a given interval that the largest root of the orginal equation is not the root of the equation with differentiated coefficients.

```
> diffCharpoly := diff(re,d)*y^2 + diff(se,d)*y + diff(te,d);
```

$$diffCharpoly := \left(2\,d - \frac{3}{2}\right) y^2 + \left(\frac{23}{64} - \frac{3}{8}\,d\right) y - \frac{1}{64}$$

This is just a quadratic equation and we can immediately determine when it has no roots:

```
> solve( coeff(diffCharpoly, y)^2 - 4*rem(diffCharpoly,y,y)*coeff(diffCharpoly,y^2) < 0,d);
```

$$\text{RealRange}\left(\text{Open}\left(\frac{29}{72}\right), \text{Open}\left(\frac{5}{8}\right)\right)$$

We consider the interval 5/8..1;

On this interval there is in fact a point where the magnitude of the largest root of the characteristic polynomial is maximal. It is useful to find it more precisely. We use Groebner bases package to eliminate y from the system of two equations and find a polynomial equation for d. **finduni** finds the minimal univariate polynomial in the ideal generated by the two polynomials:

```
> dEquation := finduni( d, [subs( lambda = y, ButterCharpoly), diffCharpoly]);
```

$$dEquation := 9 - 48\,d + 332\,d^2 - 960\,d^3 + 1152\,d^4 - 512\,d^5$$

Now find a rational interval for d; **realroot** provides us with guaranteed bounds on all real roots. Luckily, there is a single real root:

```
> IntervalDominantMax := op(realroot( dEquation, 1/10^7));
```

$$IntervalDominantMax := \left[\frac{11341469}{16777216}, \frac{5670735}{8388608}\right]$$

```
> map(evalf, IntervalDominantMax);
```

[.67600423097610473633, .67600429058074951172]

Evaluating the derivative at a point, we get a negative value; we conclude that the derivative is negative for d greater than the value above.
We use the algebraic value of the root returned by solve, rather than transcedental equation, because interval inapply does not work properly for hyperbolic functions.

```
> IntervDeriv := inapply( diff(op(1, [solve( ButterCharpoly, lambda)]),d),d): eval(IntervDeriv(0.9));
```

[-.77122129552760084746, -.77122129552759906668]

We have shown that the magnitude of the largest eigenvalue decreases from approximately 0.6760043 to 1.

Valence 3

In this case eigenvalue 1/4 is the largest and has three identical Jordan blocks.

Block 0

```
> jordan(subs( cs = 1, eval(Butter00re)));
```

$$\begin{bmatrix} \frac{1}{4} & 1 & 0 \\ 0 & \frac{1}{4} & 1 \\ 0 & 0 & \frac{1}{4} \end{bmatrix}$$

Block 1

```
> jordan( subs( cs = 1/2, eval(Butter00re)));
```

$$\begin{bmatrix} \frac{1}{16} & 0 & 0 \\ 0 & \frac{1}{4} & 1 \\ 0 & 0 & \frac{1}{4} \end{bmatrix}$$

Block 2

```
> jordan( subs( cs = -1/2, eval(Butter00re)));
```

$$\begin{bmatrix} \frac{1}{16} & 0 & 0 \\ 0 & \frac{1}{4} & 1 \\ 0 & 0 & \frac{1}{4} \end{bmatrix}$$

Valences 4,5

K = 4; Check which formulas to use.

```
> inapply(DiscrRoot); inapply(cos(Pi/7)^4); inapply(cos(2*Pi/4)^2);
```

( ) → [.84868120391246120009, .84868120391246120352]

( ) → [.65892978418482746461, .65892978418482746486]

0

The largest eigenvalue is in the first block.

```
> inapply(subs( d = cos(Pi/4)^2, AbsDominantEV1));
```

( ) → [.46904152209925298145, .46904152209925298172]

```
> inapply(subs( d = cos(Pi/2)^2,AbsDominantEV1 ));
```

( ) → [.12499999999999999998, .12500000000000000002]

K = 5;

```
> inapply(DiscrRoot); inapply(cos(Pi/5)^2); inapply(cos(2*Pi/5)^2);
```

( ) → [.84868120391246120009, .84868120391246120352]

( ) → [.65450849718747371183, .65450849718747371227]

( ) → [.09549150281252628866, .09549150281252628029]

```
> inapply(subs( d = cos(Pi/5)^2, AbsDominantEV1));
```

( ) → [.50667561139380648476, .50667561139380649194]

We do not have to check the eigenvalue of the second block: for it, d < 1/4, therefore, the magnitude of the largest eigenvalue is also less than 1/4.

For valence greater than 7, the eigenvalue of the block with m = 1 is not the largest

We have established that the magnitude of the largest eigenvalue decreases as the function of d when d > 0.6700423 =d_0; if $d = \cos\left(\frac{m\,\pi}{K}\right)^2 > d_0$

for $m = 2$, for K > 6 we can conclude that the eigenvalue for $m = 2$ is greater than the eigenvalue for $m = 1$. This is the case for K > 10:

```
>  x := inapply(2*Pi/K, K): eval(Interval_Integerpower(Interval_cos(x(11)),2));
```

[.70770750650094321267, .70770750650094321286]

For values between 7 and 10 , have to check one by one.

K = 7. Check which formulas to use. Note that for m = 2 the value is below d_0, so we do not have to check the other values of m.

```
> inapply(DiscrRoot); inapply(cos(Pi/7)^2); inapply(cos(2*Pi/7)^2);
```

( ) → [.84868120391246120009, .84868120391246120352]

( ) → [.81174490092936676519, .81174490092936676535]

( ) → [.38873953302184279774, .38873953302184279798]

In this case, the largest eigenvalue is still in the first block:

```
> inapply(subs( d = cos(Pi/7)^2, AbsDominantEV1));
```

( ) → [.48191070141255089859, .48191070141255090538]

```
> inapply(subs( d = cos(2*Pi/7)^2, AbsDominantEV1));
```

( ) → [.40611653561495260677, .40611653561495260912]

K = 8

```
> inapply(DiscrRoot); inapply(cos(Pi/8)^2); inapply(cos(2*Pi/8)^2);
```

( ) → [.8486811866, .8486812214]

( ) → [.8535533896, .8535533917]

( ) → [.4999999998, .5000000002]

For $d_0 < d$, we cannot use the transcedental expression -- intpak does not have arccosh; rather than expressing it using logarithm, we use the algebraic expresssion, which works because the discriminant is positive:

```
> inapply(subs( d = cos(Pi/8)^2,op(1, [solve(ButterCharpoly, lambda)])));
```

( ) → [.46241776728303719789, .46241776728303860919]

We see that in this case the eigenvalue of the second block is larger:

```
> inapply(subs( d = cos(2*Pi/8)^2,AbsDominantEV1));
```

( ) → [.46904152209925298145, .46904152209925298172]

K = 9; same result as for 8:

```
> inapply(DiscrRoot); inapply(cos(Pi/9)^2); inapply(cos(2*Pi/9)^2);
```

( ) → [.84868120391246120009, .84868120391246120352]

( ) → [.88302222155948901753, .88302222155948901768]

( ) → [.58682408883346517429, .58682408883346517455]

```
> inapply(subs( d = cos(Pi/9)^2,op(1, [solve(ButterCharpoly, lambda)])));
```

( ) → [.44442935560347137503, .44442935560347191330]

```
> inapply(subs( d = cos(2*Pi/9)^2,AbsDominantEV1));
```

( ) → [.49729407293962378849, .49729407293962379215]

K = 10, same as for 8 and 9:

```
> inapply(DiscrRoot); inapply(cos(Pi/10)^2); inapply(cos(2*Pi/10)^2);
```

( ) → [.84868120391246120009, .84868120391246120352]

( ) → [.90450849718747371190, .90450849718747371220]

( ) → [.65450849718747371183, .65450849718747371227]

```
> inapply(subs( d = cos(Pi/10)^2,op(1, [solve(ButterCharpoly, lambda)])));
```

( ) → [.42859991276733051144, .42859991276733166709]

```
> inapply(subs( d = cos(2*Pi/10)^2,AbsDominantEV1));
```

( ) → [.50667561139380648476, .50667561139380649194]

Summary of the eigenvalue analysis

For K = 3, the largest eigenvalue is 1/4 and has multiplicity 7, with 2 blocks of size 2 and one block of size 3. For K = 4..7 the largest eigenvalue has multiplicity 2 and corresponds to the 1st and the last block. For K > 7, the largest eigenvalues are not in the 1st and last blocks.

Eigenvectors

Find the eigenvectors in two steps. Using the special structure of the matrix $\begin{bmatrix} B_{0,0} & 0 \\ B_{1,0} & B_{1,1} \end{bmatrix}$, and the fact that we are interested in the eigenvectors which are also eigenvalues of the subblock $B_{0,0}$, we find the eigenvector as $[v_0, -(B_{1,1} - \lambda I)^{(-1)} B_{1,0} v_0]$, where v_0 is the eigenvector of $B_{0,0}$. We also use the fact that in all cases of interest, the eigenvalues of $B_{0,0}$ are not eigenvalues of $B_{1,1}$.

Compute an eigenvvector of $B_{0,0}$; check first that the two second lines of the matrix are always independent; the second component of the cross product is not zero, because $\frac{1}{4} \le \lambda$:

```
> crossprod( row( evalm( subs( Butterconst, evalm(Butterfly00)) - lambda* &*() ), 2), row( evalm( Butterfly00 - lambda* &*() ), 3));
```

$$\left[-\lambda\left(\frac{1}{8}-\lambda\right), \lambda-\frac{1}{8}, -\frac{1}{16}-\frac{1}{16}c\sim-\frac{1}{16}I\,s+\lambda\left(\frac{9}{16}+\frac{7}{16}c\sim-\frac{1}{8}c\sim^2+\frac{9}{16}I\,s-\frac{1}{8}I\,c\sim\,s\right)\right]$$

Compute the first part of the vector:

```
> v0 := subs( _t[1] = 1, map(simplify, linsolve(
          submatrix ( evalm( Butterfly00- lambda*&*() ) ,2..3,1..3), [0,0] )));
>
```

$$v0 := \left[\lambda, 1, -\frac{1}{2}\frac{I\,(-I - I\,c\sim + s - 9\,s\,\lambda + 2\,s\,c\sim\,\lambda + 9\,I\,\lambda - 2\,I\,\lambda\,c\sim^2 + 7\,I\,\lambda\,c\sim)}{-1+8\,\lambda}\right]$$

```
> Butter11lambda := evalm( Butterfly11 - lambda* &*());
>
```

$$Butter11lambda := \begin{bmatrix} -\lambda & \frac{-1}{16} & -\frac{1}{16}c\sim+\frac{1}{16}I\,s \\ 0 & -\frac{1}{16}-\lambda & 0 \\ 0 & 0 & -\frac{1}{16}-\lambda \end{bmatrix}$$

```
> v1 := map( simplify, subs( { s^3 = s*(1-c^2), s^2 = 1- c^2}, map( expand, evalm( - inverse( Butter11lambda)  &* Butterfly10 &* v0
  ))));
```

$$v1 := \left[-\frac{1}{16}\frac{-3\,c\sim-1-1024\,\lambda^3+64\,\lambda^2\,c\sim^2+165\,\lambda-208\,\lambda^2\,c\sim+45\,\lambda\,c\sim+256\,\lambda^3\,c\sim-1120\,\lambda^2-10\,\lambda\,c\sim^2}{(-1+8\,\lambda)\,(1+16\,\lambda)\,\lambda},\right.$$

$$-\frac{1}{2}\frac{-79\,\lambda-128\,\lambda^2-29\,\lambda\,c\sim-32\,\lambda^2\,c\sim-61\,I\,s\,\lambda-32\,I\,s\,\lambda^2+11+5\,c\sim+7\,I\,s+18\,I\,s\,c\sim\,\lambda+14\,\lambda\,c\sim^2}{(-1+8\,\lambda)\,(1+16\,\lambda)},$$

$$\left.\frac{1}{2}\frac{-11\,c\sim+2\,c\sim^2-11\,I\,s+61\,\lambda+128\,\lambda^2\,c\sim+61\,\lambda\,c\sim+2\,I\,c\sim\,s-32\,I\,s\,c\sim\,\lambda+79\,I\,s\,\lambda+4\,c\sim^3\lambda+32\,\lambda^2+128\,I\,s\,\lambda^2+4\,I\,c\sim^2 s\,\lambda-32\,\lambda\,c\sim^2-7}{(-1+8\,\lambda)\,(1+16\,\lambda)}\right]$$

Put the two parts of the vector together and simplify notation

```
> ButterEigenvect := array( map( simplify, [seq( v0[i], i=1..3), seq(v1[i],i=1..3)])):
```

Check if the expression makes sense for K = 6

```
>  map(expand, subs( {lambda = 1/2, c = 1/2, s = sqrt(3)/2}, eval(ButterEigenvect)));
```

$$\left[\frac{1}{2}, 1, \frac{3}{4}+\frac{1}{4}I\sqrt{3}, \frac{3}{2}, \frac{5}{4}+\frac{1}{4}I\sqrt{3}, 1+\frac{1}{2}I\sqrt{3}\right]$$

Code generation

Three functions are generated (same as for other schemes):
`Float Eigenvalue(int K)` computes the eigenvalues,
`void EigenvectorReal(Float c, Float lambda, Float* EvRe)` initializes an array for the real part of the complex eigenvector, `void EigenvectorImaginary(Float c, Float* lambda, Float* EvIm)` initializes the array for the complex part.
Memory for arrays should be allocated by the calling function.
The output is written to a file; if the name is 'default', it is written to the standard output (warning: for some reason, writing to standard output is terribly slow; writing to a file and then looking at it in an editor is much more efficient. All functions use `Float` as the name of the class for the interval numbers.
It is assumed to have explicit casts from 64-bit integers, standard arithmetics operations, and macros `FR` and `Fdiv`, (see **`ConvertToFloat`** for details).

```
> OutputFile := 'butterfly.cpp':
> MakeClassHeader( OutputFile, 'Butterfly', 2,4,3, RegButterfly):
```

Code generation for eigenvalues

To show that the scheme produces C1 surfaces for valences 4,5,7, and not C1 surfaces for other valences we need expressions for eigenvalues of the first block of the DFT-transformed subdivision matrix (m = 1).
We generate two functions, one to be used for valences 4,5,7; the other for larger valences.

`ComputeEigenvalues(N,eps,fname)` Numerically compute eigenvalues for a range, and write a function with a large table into a file.
Although we have computed explicit formulas above, they are numerically unstable for d close to 1 (for large K); the simplest solution is to to precomute the largest eigenvalue numerically with verified ; we use the fact that the largest eigenvalues is in the interval 1/4 .. 1 in the range of interest. This function computes eigenvalues up to valence **`N`**, verifying that the precision is no less than **`eps`**.

```
> ### WARNING: semantics of type 'string' have changed
  ComputeEigenvalues := proc( N::integer, eps::numeric, fname::string)
    local K, intervCharpoly,  intervPi, intervd,
    expandedCharpoly, approxEV, dK;
    global ButterCharpoly;
    Digits := 25;
    intervCharpoly := inapply( ButterCharpoly, lambda, d);
    # use arccos to make sure we get an interval for Pi
    intervPi := Interval_times([2.0,2.0],Interval_arccos([0,0]));
    intervd := inapply( cos(intervPi/K)^2, K);
    fprintf(fname, 'virtual Float Eigenvalue(int K) {\n');
    fprintf(fname, ' static INTEGER64 EV[] = {\n' );
    # write  infinity, skip 1,2, write 3
    fprintf( fname, 'CONST64(1),CONST64(1),CONST64(4),\n');
    fprintf( fname, 'CONST64(0),CONST64(0),CONST64(0),\n');
    fprintf( fname, 'CONST64(0),CONST64(0),CONST64(0),\n');
    fprintf( fname, 'CONST64(1),CONST64(1),CONST64(4),\n');
```

```
    for K from 4 to N  do
      if  (K - 4) mod 100 = 0 then print(K);
      fi;
      dK := intervd(K);
      expandedCharpoly := subs( d = cos(Pi/K)^2, ButterCharpoly);
      Digits := 25;
      approxEV := fsolve( expandedCharpoly, lambda, lambda=0.25..1);
      # check that the precision is at least eps
      if  op(2, intervCharpoly(approxEV-eps, dK)) > 0
         or op(1, intervCharpoly(approxEV+eps, dK)) < 0  then
        ERROR('fsolve precision failure for K =', K);
      fi;
      Digits := 15; # need to get the right number of digits for printing
      fprintf(fname, 'CONST64(%d),CONST64(%d),CONST64(%d),\n', op(1,evalf(approxEV-eps)),
  op(1,evalf(approxEV+eps)),10^(-op(2,evalf(approxEV))));
      Digits := 25;
    od;
    fprintf(fname, 'CONST64(0)};\n return Float(EV[3*K],EV[3*K+1])/Float(EV[3*K+2]) ;\n}\n\n');
    NULL;
  end:
```

Generate eigenvalues

```
> ComputeEigenvalues( 4500, 1e-10, OutputFile):
> optEV1 := 'optimize/makeproc'(map(ConvertToFloat,[optimize( AbsDominantEV1)]),parameters=[d]):
>
```

Fix a problem with C code generation: no return value is generated, if the last statement is assignment

```
> optEV1body := procbody(optEV1):
  EVstatseq := op(5,optEV1body): EVstatseq :=  '&statseq'(op(EVstatseq), op(1,op(-1,EVstatseq))):
> optEV1 := procmake(subsop(5=EVstatseq, optEV1body)):
> S := []: C(optEV1,ansi):
  for i from 1 to vectdim(S) do
    fprintf(OutputFile, replaceall(op(i,S),'double','Float')):
  od:
```

Second version for valences >= 8;

```
> optEV2 := 'optimize/makeproc'(map(ConvertToFloat,[optimize( op(1,[solve(ButterCharpoly,lambda)]))]),parameters=[d]):
> optEV2body := procbody(optEV2):
> EVstatseq := op(5,optEV2body): EVstatseq :=  '&statseq'(op(EVstatseq), op(1,op(-1,EVstatseq))):
> optEV2 := procmake( subsop(5=EVstatseq, optEV2body)):
> S := []: C(optEV2,ansi):
  for i from 1 to vectdim(S) do
    fprintf(OutputFile, replaceall(op(i,S),'double','Float')):
  od:
>
```

This function is used to construct interval eigenvectors "at infinity"; assume d sufficiently close to 1, so that $\hat{\lambda}(c)$ decreases.

```
> fprintf( OutputFile,  'virtual void EigenvalueRange( Float c, Float& lambdamin, Float& lambdamax) {\n'):
  fprintf( OutputFile,  'Float d = FR(1,2)*(c + Float(1)); assert( (exactfloor(d)).as_double() > 0.676);\n'):
  fprintf( OutputFile,  'if( exactceil(c)  < exactfloor( sqrt(Float(2))/Float(2)) )\n'):
  fprintf( OutputFile,  'lambdamax = optEV1(d);\n  else lambdamax = optEV2(d); lambdamin = FR(1,4);\n}\n\n'):
>
```

Code generation for eigenvectors

Two functions are generated; one initializes an array for the real part of the complex eigenvector, the other for the complex part. Memory should be allocated by the calling function.

```
> GenerateEigenvectorCode(ButterEigenvect, OutputFile);
> fprintf(OutputFile, '};'):
> fclose(OutputFile);
```

Modified Butterfly scheme

The Modified Butterfly scheme by construction always has the subdominant eigenvalue 1/2 in the first block of the DFT-transformed subdivision matrix.
Thus, we only need to compute the eigenvectors for the characteristic map analysis. We do not assume that w = 1/16 here.

Compute the complex eigenvector

```
> ModButter := matrix( [
    [ 1/2, 0, 0,0,0,0],
    [1,0,0,0,0,0],
    [(1/2)*(1+ omega) - w*(conjugate(omega) + omega^2 ), -w*(1 + omega),2*w,0,0,0],
    [1/2 - 2*w*c,1/2,2*w*(1+conjugate(omega)),0, -w, -w*conjugate(omega)],
    [1/2 + 2*w*omega, 2*w - w*omega,1/2-w*conjugate(omega),0,-w,0],
    [(1/2)*omega + 2*w,2*w*omega-w, 1/2 - w*omega,0,0,-w]]);
```

$$ModButter := \begin{bmatrix} \frac{1}{2} & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{2}+\frac{1}{2}\omega - w(\overline{\omega}+\omega^2) & -w(1+\omega) & 2w & 0 & 0 & 0 \\ \frac{1}{2}-2wc\sim & \frac{1}{2} & 2w(\overline{\omega}+1) & 0 & -w & -w\overline{\omega} \\ \frac{1}{2}+2w\omega & 2w-w\omega & \frac{1}{2}-w\overline{\omega} & 0 & -w & 0 \\ \frac{1}{2}\omega+2w & 2w\omega-w & \frac{1}{2}-w\omega & 0 & 0 & -w \end{bmatrix}$$

This is simple enough for the Maple function.

```
> jordan(ModButter,'P');
```

$$\begin{bmatrix} \frac{1}{2} & 0 & 0 & 0 & 0 & 0 \\ 0 & -w & 0 & 0 & 0 & 0 \\ 0 & 0 & 2\,w & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -w \end{bmatrix}$$

```
> ModButterEigenvect := subs( { w = 1/16, cos(2*m*Pi/K) = c, sin(2*m*Pi/K) = s}, map( simplify, map( evalc, subs( Buttervar,
  col(eval('P'),1)))));
```

ModButterEigenvect :=

$$\left[1,\ 2,\ -\frac{1}{3}c\sim^2+\frac{7}{6}+\frac{5}{6}c\sim-\frac{1}{3}I\,c\sim\,s+\frac{7}{6}I\,s,\ \frac{653}{216}-\frac{11}{108}c\sim^2+\frac{1}{216}c\sim,\ \frac{35}{54}c\sim-\frac{7}{27}c\sim^2+\frac{121}{54}+\frac{7}{6}I\,s-\frac{1}{3}I\,c\sim\,s,\ \frac{2}{27}c\sim^3+\frac{103}{54}c\sim-\frac{14}{27}c\sim^2+\frac{7}{6}+\frac{2}{27}I\,s\,c\sim^2+\frac{121}{54}I\,s-\frac{14}{27}I\,c\sim\,s\right]$$

Code generation