# Coordinate mapping

Work in progress

```
PeS = PowerExpand[Simplify[##]] &
FpPeS = FixedPoint[PeS, ##] &
```

```
PowerExpand[Simplify[##1]] &
```

```
FixedPoint[PeS, ##1] &
```

```
FsSubst[rules_] := FullSimplify[# //. rules] &
FpFsSubst[rules_] := FixedPoint[FsSubst[rules], #] &
```

```
LSolve = Last[Solve[##]] &
```

```
Last[Solve[##1]] &
```

## Coordinate transformation for the muralizer

We want to map Cartesian coordinates $\vec{\boldsymbol{x}} \equiv \begin{pmatrix} x \\ y \end{pmatrix}$ into radii $\begin{pmatrix} r_A \\ r_B \end{pmatrix}$ from two reference points, $\vec{\boldsymbol{A}} \equiv \begin{pmatrix} x_A \\ y_A \end{pmatrix}$, $\vec{\boldsymbol{B}} \equiv \begin{pmatrix} x_B \\ y_B \end{pmatrix}$.

$$\begin{pmatrix} r_A \\ r_B \end{pmatrix} = \begin{pmatrix} \sqrt{(x - x_A)^2 + (y - y_A)^2} \\ \sqrt{(x - x_B)^2 + (y - y_B)^2} \end{pmatrix} = \sqrt{\begin{pmatrix} \left\langle \vec{\boldsymbol{x}} - \vec{\boldsymbol{A}} \,\middle|\, \vec{\boldsymbol{x}} - \vec{\boldsymbol{A}} \right\rangle \\ \left\langle \vec{\boldsymbol{x}} - \vec{\boldsymbol{B}} \,\middle|\, \vec{\boldsymbol{x}} - \vec{\boldsymbol{B}} \right\rangle \end{pmatrix}}$$

```
stringcoordinates[refpoints_] := Function[{x}, Sqrt[#.#] &[x - #] & /@ refpoints]
```

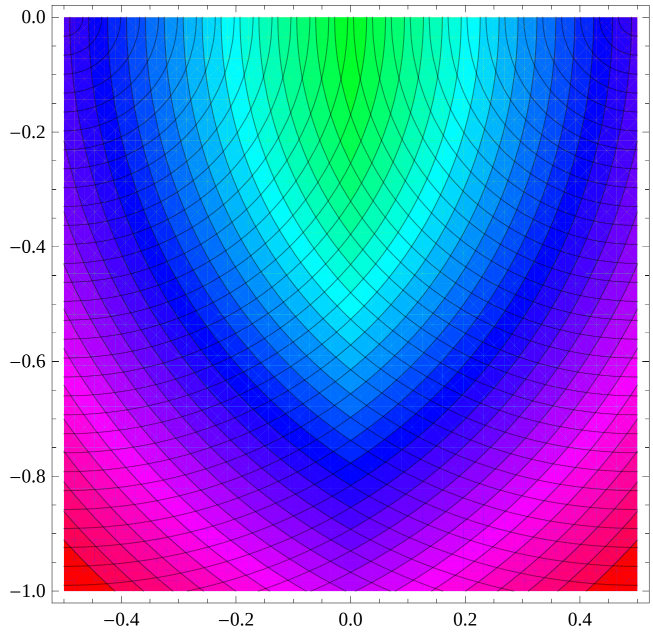
```
refpoints = {{-1/2, 0}, {1/2, 0}}
```

$$\left\{\left\{-\frac{1}{2}, 0\right\}, \left\{\frac{1}{2}, 0\right\}\right\}$$

```
stringcoordinates[refpoints][{x, y}]
```

$$\left\{\sqrt{\left(\frac{1}{2} + x\right)^2 + y^2}, \sqrt{\left(-\frac{1}{2} + x\right)^2 + y^2}\right\}$$

```
ContourPlot[stringcoordinates[refpoints][{x, y}],
 {x, -1/2, 1/2}, {y, -1, 0}, ColorFunction → Hue, Contours → 41]
```




## Transformation of a straight line

### Examples

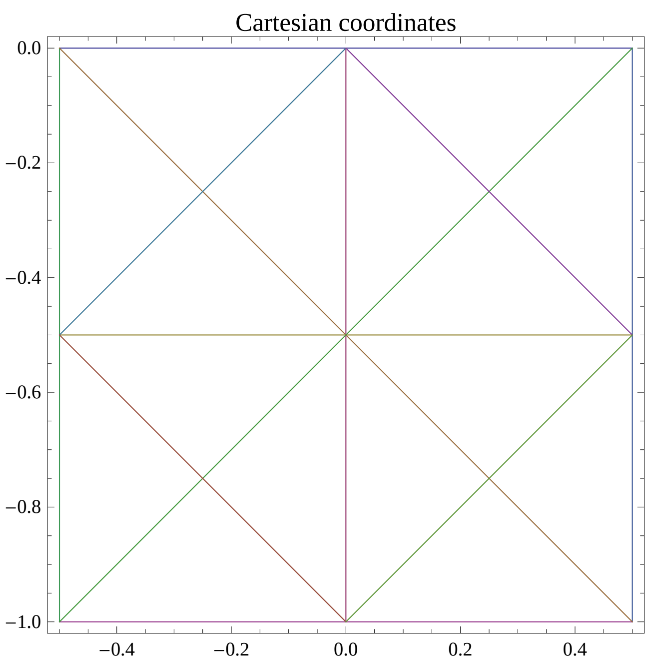
```
refpoints
```

$\left\{\left\{-\frac{1}{2}, 0\right\}, \left\{\frac{1}{2}, 0\right\}\right\}$

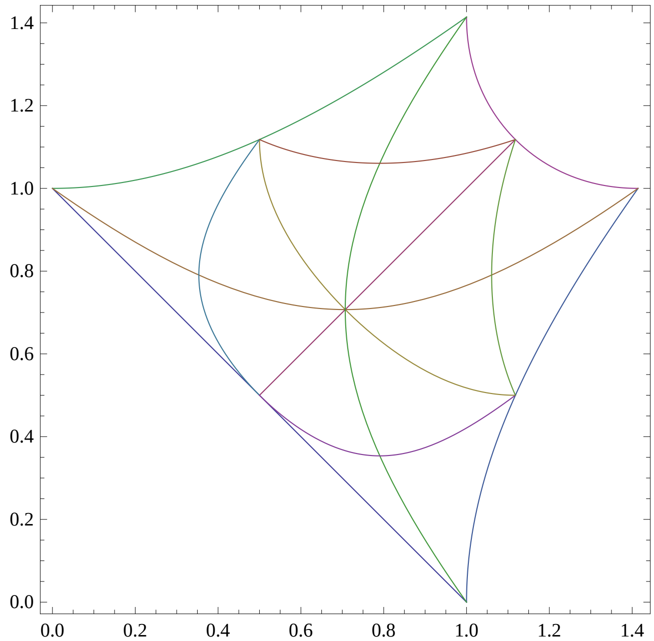
```
TableForm[lineset = {
    {{-0.5, 0}, {0.5, 0}},
    {{0, 0}, {0, -1}},
    {{-0.5, -0.5}, {0.5, -0.5}},
    {{-0.5, 0}, {-0.5, -1}},
    {{0.5, 0}, {0.5, -1}},
    {{-0.5, -1}, {0.5, -1}},
    {{-0.5, 0}, {0.5, -1}},
    {{0.5, 0}, {-0.5, -1}},
    {{-0.5, -0.5}, {0, 0}},
    {{0.5, -0.5}, {0, 0}},
    {{-0.5, -0.5}, {0, -1}},
    {{0.5, -0.5}, {0, -1}}
  }, TableDepth → 2]
```

| | |
|---|---|
| {-0.5, 0} | {0.5, 0} |
| {0, 0} | {0, -1} |
| {-0.5, -0.5} | {0.5, -0.5} |
| {-0.5, 0} | {-0.5, -1} |
| {0.5, 0} | {0.5, -1} |
| {-0.5, -1} | {0.5, -1} |
| {-0.5, 0} | {0.5, -1} |
| {0.5, 0} | {-0.5, -1} |
| {-0.5, -0.5} | {0, 0} |
| {0.5, -0.5} | {0, 0} |
| {-0.5, -0.5} | {0, -1} |
| {0.5, -0.5} | {0, -1} |

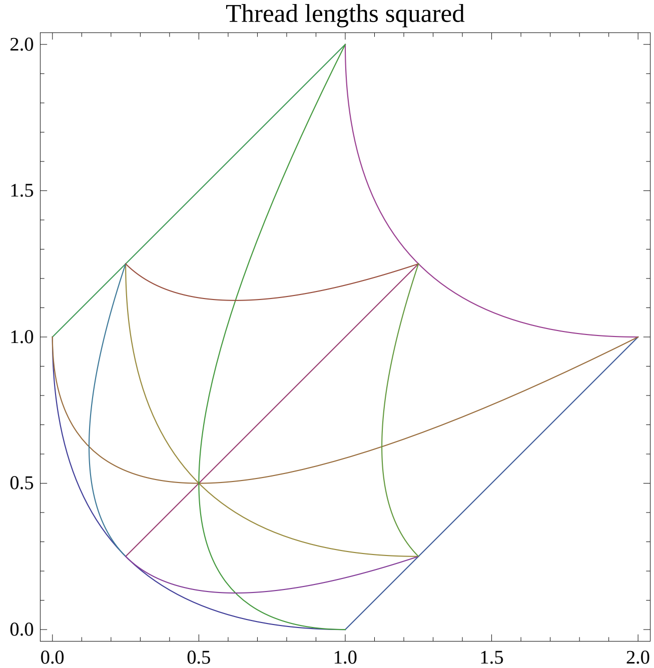
```
Print /@ {
   ParametricPlot[Evaluate[#[[1]] (1 - t) + #[[2]] t & /@ #], {t, 0, 1}, PlotRange → All,
      Frame → True, Axes → False, PlotLabel → "Cartesian coordinates"] &[lineset],
   ParametricPlot[Evaluate[stringcoordinates[refpoints][#[[1]] (1 - t) + #[[2]] t] & /@ #],
      {t, 0, 1}, PlotRange → All, Frame → True, PlotLabel → "Thread lengths"] &[lineset],
   ParametricPlot[Evaluate[stringcoordinates[refpoints][#[[1]] (1 - t) + #[[2]] t]^2 & /@ #], {t,
       0, 1}, PlotRange → All, Frame → True, PlotLabel → "Thread lengths squared"] &[lineset]
  };
```



Thread lengths




Thread lengths squared




## ■ Mathematics

$$\texttt{Simplify}\left[\{x1, y1\}\ (1-t) + \{x2, y2\}\ t\ /.\ \left\{x1 \to x - \frac{\delta x}{2},\ y1 \to y - \frac{\delta y}{2},\ x2 \to x + \frac{\delta x}{2},\ y2 \to y + \frac{\delta y}{2}\right\}\right]$$

$$\left\{x + \left(-\frac{1}{2} + t\right) \delta x,\ y + \left(-\frac{1}{2} + t\right) \delta y\right\}$$

as implicit function: a quadratic form in $\{R, S\}$

```
{R, S} == FullSimplify[
  {#.# &[{Xx, Yy} - {XA, YA}], #.# &[{Xx, Yy} - {XB, YB}]} /. {XA → X0 - ΔX/2, YA → Y0 - ΔY/2,
     XB → X0 + ΔX/2, YB → Y0 + ΔY/2} /. {Xx → x + (t - 1/2) δx, Yy → y + (t - 1/2) δy}]
Eliminate[%, t] /. {(A_ == B_) → (A - B == 0)}
Collect[First[%], {R, S}, Simplify]
TableForm[coefficients = FullSimplify[LSolve[% == a R^2 + b S^2 + c R S + d R + e S + f /.
      {{R → 0, S → 0}, {R → 1, S → 0}, {R → 1, S → 1}, {R → 0, S → 1}, {R → -1, S → -1},
       {R → 0, S → -1}, {R → -1, S → 0}, {R → 200, S → -42}}, {a, b, c, d, e, f}]]
]
```

$$\{R, S\} == \left\{\left(x - X0 + \left(-\frac{1}{2} + t\right)\delta x + \frac{\Delta X}{2}\right)^2 + \left(y - Y0 + \left(-\frac{1}{2} + t\right)\delta y + \frac{\Delta Y}{2}\right)^2,\ \frac{1}{4}\left((-2\,x + 2\,X0 + \delta x - 2\,t\,\delta x + \Delta X)^2 + (-2\,y + 2\,Y0 + \delta y - 2\,t\,\delta y + \Delta Y)^2\right)\right\}$$

$$S^2\,\delta x^2 - 2\,S\,\delta x^2\,\Delta X^2 + 4\,y^2\,\delta x^2\,\Delta X^2 - 8\,y\,Y0\,\delta x^2\,\Delta X^2 + 4\,Y0^2\,\delta x^2\,\Delta X^2 + \delta x^2\,\Delta X^4 - 4\,S\,y\,\delta x\,\Delta X\,\delta y + 4\,S\,Y0\,\delta x\,\Delta X\,\delta y - 8\,x\,y\,\delta x\,\Delta X^2\,\delta y + 8\,X0\,y\,\delta x\,\Delta X^2\,\delta y + 8\,x\,Y0\,\delta x\,\Delta X^2\,\delta y - 8\,X0\,Y0\,\delta x\,\Delta X^2\,\delta y + S^2\,\delta y^2 + 4\,S\,x\,\Delta X\,\delta y^2 - 4\,S\,X0\,\Delta X\,\delta y^2 + 4\,x^2\,\Delta X^2\,\delta y^2 - 8\,x\,X0\,\Delta X^2\,\delta y^2 + 4\,X0^2\,\Delta X^2\,\delta y^2 + R^2\left(\delta x^2 + \delta y^2\right) + 4\,S\,y\,\delta x^2\,\Delta Y - 4\,S\,Y0\,\delta x^2\,\Delta Y - 4\,S\,x\,\delta x\,\delta y\,\Delta Y + 4\,S\,X0\,\delta x\,\delta y\,\Delta Y - 4\,S\,\delta x\,\Delta X\,\delta y\,\Delta Y + 2\,\delta x\,\Delta X^3\,\delta y\,\Delta Y + 4\,y^2\,\delta x^2\,\Delta Y^2 - 8\,y\,Y0\,\delta x^2\,\Delta Y^2 + 4\,Y0^2\,\delta x^2\,\Delta Y^2 + \delta x^2\,\Delta X^2\,\Delta Y^2 - 8\,x\,y\,\delta x\,\delta y\,\Delta Y^2 + 8\,X0\,y\,\delta x\,\delta y\,\Delta Y^2 + 8\,x\,Y0\,\delta x\,\delta y\,\Delta Y^2 - 8\,X0\,Y0\,\delta x\,\delta y\,\Delta Y^2 - 2\,S\,\delta y^2\,\Delta Y^2 + 4\,x^2\,\delta y^2\,\Delta Y^2 - 8\,x\,X0\,\delta y^2\,\Delta Y^2 + 4\,X0^2\,\delta y^2\,\Delta Y^2 + \Delta X^2\,\delta y^2\,\Delta Y^2 + 2\,\delta x\,\Delta X\,\delta y\,\Delta Y^3 + \delta y^2\,\Delta Y^4 + R\left(-2\,S\,\delta x^2 - 2\,\delta x^2\,\Delta X^2 + 4\,y\,\delta x\,\Delta X\,\delta y - 4\,Y0\,\delta x\,\Delta X\,\delta y - 2\,S\,\delta y^2 - 4\,x\,\Delta X\,\delta y^2 + 4\,X0\,\Delta X\,\delta y^2 - 4\,y\,\delta x^2\,\Delta Y + 4\,Y0\,\delta x^2\,\Delta Y + 4\,x\,\delta x\,\delta y\,\Delta Y - 4\,X0\,\delta x\,\delta y\,\Delta Y - 4\,\delta x\,\Delta X\,\delta y\,\Delta Y - 2\,\delta y^2\,\Delta Y^2\right) == 0$$

$$R^2\left(\delta x^2 + \delta y^2\right) + S^2\left(\delta x^2 + \delta y^2\right) + \left(\Delta X^2 + \Delta Y^2\right)\left(4\,y^2\,\delta x^2 + 4\,Y0^2\,\delta x^2 + \delta x^2\,\Delta X^2 + 8\,(x - X0)\,Y0\,\delta x\,\delta y + 4\,x^2\,\delta y^2 - 8\,x\,X0\,\delta y^2 + 4\,X0^2\,\delta y^2 - 8\,y\,\delta x\,(Y0\,\delta x + (x - X0)\,\delta y) + 2\,\delta x\,\Delta X\,\delta y\,\Delta Y + \delta y^2\,\Delta Y^2\right) - 2\,S\left(\delta x^2\left(\Delta X^2 + 2\,(-y + Y0)\,\Delta Y\right) + 2\,\delta x\,\delta y\,(y\,\Delta X - Y0\,\Delta X + (x - X0 + \Delta X)\,\Delta Y) + \delta y^2\left(-2\,x\,\Delta X + 2\,X0\,\Delta X + \Delta Y^2\right)\right) + R\left(-2\,S\left(\delta x^2 + \delta y^2\right) - 2\left(\delta x^2\left(\Delta X^2 + 2\,(y - Y0)\,\Delta Y\right) + 2\,\delta x\,\delta y\,(-y\,\Delta X + Y0\,\Delta X + (-x + X0 + \Delta X)\,\Delta Y) + \delta y^2\left(2\,x\,\Delta X - 2\,X0\,\Delta X + \Delta Y^2\right)\right)\right)$$

$$a \to \delta x^2 + \delta y^2$$
$$b \to \delta x^2 + \delta y^2$$
$$c \to -2\left(\delta x^2 + \delta y^2\right)$$
$$d \to -2\left(\delta x^2\left(\Delta X^2 + 2\,(y - Y0)\,\Delta Y\right) + 2\,\delta x\,\delta y\,(-y\,\Delta X + Y0\,\Delta X + (-x + X0 + \Delta X)\,\Delta Y) + \delta y^2\left(2\,x\,\Delta X - 2\,X0\,\Delta X + \Delta Y^2\right)\right)$$
$$e \to -2\left(\delta x^2\left(\Delta X^2 + 2\,(-y + Y0)\,\Delta Y\right) + 2\,\delta x\,\delta y\,(y\,\Delta X - Y0\,\Delta X + (x - X0 + \Delta X)\,\Delta Y) + \delta y^2\left(-2\,x\,\Delta X + 2\,X0\,\Delta X + \Delta Y^2\right)\right)$$
$$f \to \left(\Delta X^2 + \Delta Y^2\right) \big(4\,y^2\,\delta x^2 + 4\,Y0^2\,\delta x^2 + 8\,(x - X0)\,Y0\,\delta x\,\delta y + 4\,(x - X0)^2\,\delta y^2 - 8\,y\,\delta x\,(Y0\,\delta x + (x - X0)\,\delta y) + (\delta x\,\Delta X$$

```
TableForm[FpFsSubst[{δx^2 + δy^2 → δ^2, ΔX^2 + ΔY^2 → Δ^2, x → X0 + ξ, y → Y0 + υ}][coefficients]]
Simplify[{(d + e)/2, (d - e)/2} /. %]
```

$$
\begin{array}{l}
a \to \delta^2 \\
b \to \delta^2 \\
c \to -2\,\delta^2 \\
d \to -2\left(\delta y^2 \left(\Delta Y^2 + 2\,\Delta X\,\xi\right) + \delta x^2 \left(\Delta X^2 + 2\,\Delta Y\,\upsilon\right) - 2\,\delta x\,\delta y\,(\Delta Y\,\xi + \Delta X\,(-\Delta Y + \upsilon))\right) \\
e \to -2\left(\delta y^2 \left(\Delta Y^2 - 2\,\Delta X\,\xi\right) + \delta x^2 \left(\Delta X^2 - 2\,\Delta Y\,\upsilon\right) + 2\,\delta x\,\delta y\,(\Delta Y\,\xi + \Delta X\,(\Delta Y + \upsilon))\right) \\
f \to \Delta^2 \left(\delta y^2 \left(\Delta Y^2 + 4\,\xi^2\right) + 2\,\delta x\,\delta y\,(\Delta X\,\Delta Y - 4\,\xi\,\upsilon) + \delta x^2 \left(\Delta X^2 + 4\,\upsilon^2\right)\right)
\end{array}
$$

$$\left\{-2\,(\delta x\,\Delta X + \delta y\,\Delta Y)^2,\ -4\,(\Delta X\,\delta y - \delta x\,\Delta Y)\,(\delta y\,\xi - \delta x\,\upsilon)\right\}$$

```
TableForm[({a, b, c, d, e, f} /. coefficients)]
```

$$
\begin{array}{l}
\delta x^2 + \delta y^2 \\
\delta x^2 + \delta y^2 \\
-2\left(\delta x^2 + \delta y^2\right) \\
-2\left(\delta x^2 \left(\Delta X^2 + 2\,(y - Y0)\,\Delta Y\right) + 2\,\delta x\,\delta y\,(-y\,\Delta X + Y0\,\Delta X + (-x + X0 + \Delta X)\,\Delta Y) + \delta y^2 \left(2\,x\,\Delta X - 2\,X0\,\Delta X + \Delta Y^2\right)\right) \\
-2\left(\delta x^2 \left(\Delta X^2 + 2\,(-y + Y0)\,\Delta Y\right) + 2\,\delta x\,\delta y\,(y\,\Delta X - Y0\,\Delta X + (x - X0 + \Delta X)\,\Delta Y) + \delta y^2 \left(-2\,x\,\Delta X + 2\,X0\,\Delta X + \Delta Y^2\right)\right) \\
\left(\Delta X^2 + \Delta Y^2\right) \left(4\,y^2\,\delta x^2 + 4\,Y0^2\,\delta x^2 + 8\,(x - X0)\,Y0\,\delta x\,\delta y + 4\,(x - X0)^2\,\delta y^2 - 8\,y\,\delta x\,(Y0\,\delta x + (x - X0)\,\delta y) + (\delta x\,\Delta X + \delta
\end{array}
$$

```
TableForm[Simplify[
  {δx^2 + δy^2,
    δx^2 + δy^2,
    -2 (δx^2 + δy^2),
    -2 (δx ΔX + δy ΔY)^2 - 4 (ΔX δy - δx ΔY) (δy (x - X0) - δx (y - Y0)),
    -2 (δx ΔX + δy ΔY)^2 + 4 (ΔX δy - δx ΔY) (δy (x - X0) - δx (y - Y0)),
    (ΔX^2 + ΔY^2) ((δx ΔX + δy ΔY)^2 + 4 (δy (x - X0) - δx (y - Y0))^2)} -
   ({a, b, c, d, e, f} /. coefficients)]]
```

$$
\begin{array}{l}
0 \\
0 \\
0 \\
0 \\
0 \\
0
\end{array}
$$

## Implicit equation for thread lengths squared to draw a straight line

To draw a straight line from $\begin{pmatrix} x_1 \\ y_1 \end{pmatrix}$ to $\begin{pmatrix} x_2 \\ y_2 \end{pmatrix}$ in Cartesian coordinates, the thread lengths $\{r, s\}$ of the spools suspended from the points $\left\{\begin{pmatrix} X_a \\ Y_a \end{pmatrix}, \begin{pmatrix} X_b \\ Y_b \end{pmatrix}\right\}$ follow the implicit equation

$$\begin{pmatrix} r^2 & s^2 \end{pmatrix} \cdot \begin{pmatrix} A & C \\ C & B \end{pmatrix} \cdot \begin{pmatrix} r^2 \\ s^2 \end{pmatrix} + \begin{pmatrix} r^2 & s^2 \end{pmatrix} \begin{pmatrix} D \\ E \end{pmatrix} + F = 0$$

with

$A \equiv B \equiv -C \equiv \delta x^2 + \delta y^2$

$D \equiv -2\,(\delta x\,\Delta X + \delta y\,\Delta Y)^2 - 4\,(\Delta X\,\delta y - \delta x\,\Delta Y)\,(\delta y\,(x - X_0) - \delta x\,(y - Y_0))$

$E \equiv -2\,(\delta x\,\Delta X + \delta y\,\Delta Y)^2 + 4\,(\Delta X\,\delta y - \delta x\,\Delta Y)\,(\delta y\,(x - X_0) - \delta x\,(y - Y_0))$

$F \equiv \left(\Delta X^2 + \Delta Y^2\right)\left((\delta x\,\Delta X + \delta y\,\Delta Y)^2 + 4\,(\delta y\,(x - X_0) - \delta x\,(y - Y_0))^2\right)$

where

$X_0 \equiv \frac{X_a + X_b}{2}$,

$Y_0 \equiv \frac{Y_a + Y_b}{2}$,

$\Delta X \equiv X_b - X_a$,

$\Delta Y \equiv Y_b - Y_a$,

$x \equiv \frac{x_1 + x_2}{2}$,

$y \equiv \frac{y_1 + y_2}{2}$,

$\delta x \equiv x_2 - x_1$,

$\delta y \equiv y_2 - y_1$.

### Sanity check

```
Chop[
 Simplify[(((Function[{r, s}, Evaluate[Simplify[{R, S}.{{a, c/2}, {c/2, b}}.{R, S} + {R, S}.{d, e} +
          f /. coefficients /. {ΔX → XB - XA, ΔY → YB - YA, X0 → (XA + XB)/2, Y0 → (YA + YB)/2, δx →
          x2 - x1, δy → y2 - y1, x → (x1 + x2)/2, y → (y1 + y2)/2} /. {R → r^2, S → s^2}]]
      ] /. {XA → refpoints⟦1, 1⟧, YA → refpoints⟦1, 2⟧, XB → refpoints⟦2, 1⟧,
      YB → refpoints⟦2, 2⟧} /. {x1 → #⟦1, 1⟧, y1 → #⟦1, 2⟧, x2 → #⟦2, 1⟧, y2 → #⟦2, 2⟧}) @@
    (stringcoordinates[refpoints][#⟦1⟧ (1 - t) + #⟦2⟧ t])) & /@ lineset]]
```

```
{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}
```

## Inverse transformation

$$r_A{}^2 = (x - x_A)^2 + (y - y_A)^2 = x^2 - 2\,x\,x_A + x_A{}^2 + y^2 - 2\,y\,y_A + y_A{}^2$$
$$r_B{}^2 = (x - x_B)^2 + (y - y_B)^2 = x^2 - 2\,x\,x_B + x_B{}^2 + y^2 - 2\,y\,y_B + y_B{}^2$$

```
FpPeS[Solve[{r, s} == stringcoordinates[{{A, C}, {B, D}}][{x, y}], {x, y}]];
FullSimplify[
  % //. {A^2 - 2 A B + B^2 → (A - B)^2, A^2 C - 2 A B C + B^2 C → C (A - B)^2, A^2 D - 2 A B D + B^2 D → D (A - B)^2,
    C^2 - 2 C D + D^2 → (C - D)^2, C^4 - 4 C^3 D + 6 C^2 D^2 - 4 C D^3 + D^4 → (C - D)^4, -2 C^2 r^2 + 4 C D r^2 - 2 D^2 r^2 →
     -2 (C - D)^2 r^2, -2 C^2 s^2 + 4 C D s^2 - 2 D^2 s^2 → -2 (C - D)^2 s^2, r^4 - 2 r^2 s^2 + s^4 → (r^2 - s^2)^2}];
FullSimplify[% //. {A^2 (C + D) - 2 A B (C + D) + B^2 (C + D) → (A - B)^2 (C + D),
    (C - D) (C^2 - D^2 - r^2 + s^2) → (C - D)^2 (C + D) - (C - D) (r^2 - s^2), (A - B)^2 (C + D) + (C - D)^2 (C + D) →
     ((A - B)^2 + (C - D)^2) (C + D), A (-B^2 + (C - D)^2 - r^2 + s^2) → A (C - D)^2 - A B^2 - A (r - s) (r + s),
    B ((C - D)^2 + (r - s) (r + s)) → B (C - D)^2 + B (r - s) (r + s), A (C - D)^2 + B (C - D)^2 → (A + B) (C - D)^2,
    -A (r - s) (r + s) + B (r - s) (r + s) → -(A - B) (r - s) (r + s), A^3 - A^2 B - A B^2 + B^3 → (A - B)^2 (A + B),
    (A - B)^2 (A + B) + (A + B) (C - D)^2 → (A + B) ((A - B)^2 + (C - D)^2)}];
FullSimplify[% /. {(A - B)^2 + (C - D)^2 → Δ^2}];
FullSimplify[% /. {(B - A) → Δx, (A - B) → -Δx, (C - D) → -Δy, (r + s) → Σr, (r - s) → Δr}];
inversexform =
 % /. {Sqrt[א_] → ⅈ Sqrt[-א]} /. {((K_ + L_) Δ^2 + M_)/(2 Δ^2) → (K + L)/2 + M/(2 Δ^2)} //. {(Δ - Δr) (Δ + Δr) → Δ^2 - Δr^2,
    (Δ - Σr) (Δ + Σr) → Δ^2 - Σr^2} /. {(K_ + L_ Sqrt[M_])/(2 Δ^2) → K/(2 Δ^2) + (L Sqrt[M])/(2 Δ^2)} /. {A + B → 2 x̄, C + D → 2 ȳ}
```

$$\left\{\left\{x \to \frac{\triangle r\,\triangle x\,\Sigma r}{2\,\triangle^2} + \frac{\triangle y\sqrt{-\left(\triangle^2 - \triangle r^2\right)\left(\triangle^2 - \Sigma r^2\right)}}{2\,\triangle^2} + \overline{x},\ y \to \frac{\triangle r\,\triangle y\,\Sigma r}{2\,\triangle^2} - \frac{\triangle x\sqrt{-\left(\triangle^2 - \triangle r^2\right)\left(\triangle^2 - \Sigma r^2\right)}}{2\,\triangle^2} + \overline{y}\right\},\right.$$
$$\left.\left\{x \to \frac{\triangle r\,\triangle x\,\Sigma r}{2\,\triangle^2} - \frac{\triangle y\sqrt{-\left(\triangle^2 - \triangle r^2\right)\left(\triangle^2 - \Sigma r^2\right)}}{2\,\triangle^2} + \overline{x},\ y \to \frac{\triangle r\,\triangle y\,\Sigma r}{2\,\triangle^2} + \frac{\triangle x\sqrt{-\left(\triangle^2 - \triangle r^2\right)\left(\triangle^2 - \Sigma r^2\right)}}{2\,\triangle^2} + \overline{y}\right\}\right\}$$

```
cartesiancoordinates[refpoints_] :=
 Function[{R}, Block[{ra = R[[1]], rb = R[[2]], xa = refpoints[[1, 1]],
    ya = refpoints[[1, 2]], xb = refpoints[[2, 1]], yb = refpoints[[2, 2]]},
   Evaluate[{x, y} /. First[inversexform] //. {Σr → ra + rb, Δr → ra - rb, Δx → xb - xa,
       Δy → yb - ya, x̄ → (xa + xb)/2, ȳ → (ya + yb)/2, Δ^2 → Δx^2 + Δy^2, 1/Δ^2 → 1/(Δx^2 + Δy^2)}]
  ]
 ]
```

```
cartesiancoordinates[{{xa, ya}, {xb, yb}}][{ra, rb}]
```

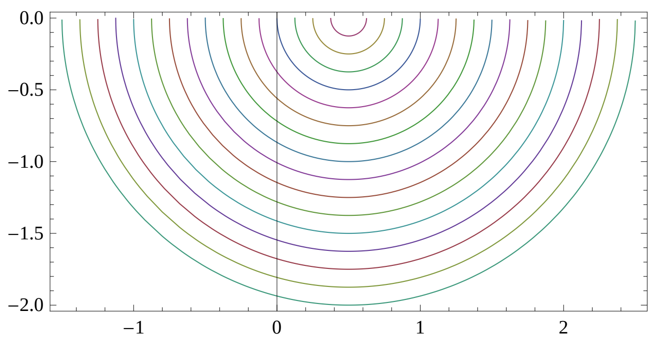
$$\left\{\frac{xa+xb}{2}+\frac{(ra-rb)\,(ra+rb)\,(-xa+xb)}{2\left((-xa+xb)^2+(-ya+yb)^2\right)}+\frac{(-ya+yb)\sqrt{-\left(-(ra-rb)^2+(-xa+xb)^2+(-ya+yb)^2\right)\left(-(ra+rb)^2+(-xa+xb)^2+(-ya+yb)^2\right)}}{2\left((-xa+xb)^2+(-ya+yb)^2\right)},\right.$$
$$\left.\frac{ya+yb}{2}+\frac{(ra-rb)\,(ra+rb)\,(-ya+yb)}{2\left((-xa+xb)^2+(-ya+yb)^2\right)}-\frac{(-xa+xb)\sqrt{-\left(-(ra-rb)^2+(-xa+xb)^2+(-ya+yb)^2\right)\left(-(ra+rb)^2+(-xa+xb)^2+(-ya+yb)^2\right)}}{2\left((-xa+xb)^2+(-ya+yb)^2\right)}\right\}$$

- **Trajectories of straight lines in the radii coordinate system**

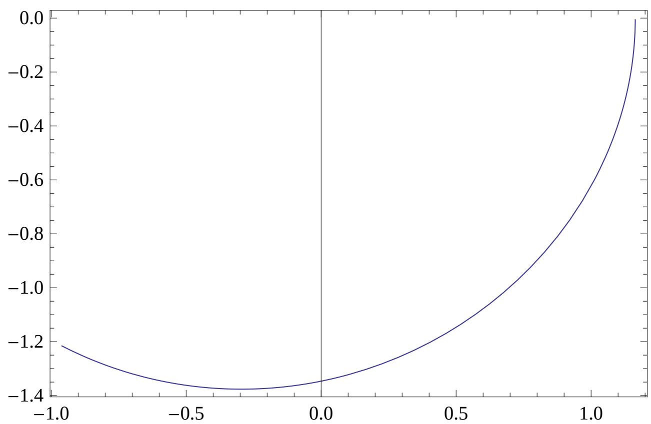
```
cartesiancoordinates[refpoints][{ra, rb}]
```

$$\left\{\frac{1}{2}(ra-rb)\,(ra+rb),\ -\frac{1}{2}\sqrt{-\left(1-(ra-rb)^2\right)\left(1-(ra+rb)^2\right)}\right\}$$

```
ParametricPlot[Evaluate[Table[cartesiancoordinates[refpoints][{ra, rb}], {rb, 0, 2, 1/8}]],
 {ra, 0, 4}, Frame → True]
```




```
ParametricPlot[Evaluate[cartesiancoordinates[refpoints][{ra, rb}] /.
   {ra → 1.3 (1 - r) + 1.8 r, rb → 1.9 (1 - r) + .2 r}], {r, 0, 1}, Frame → True]
```



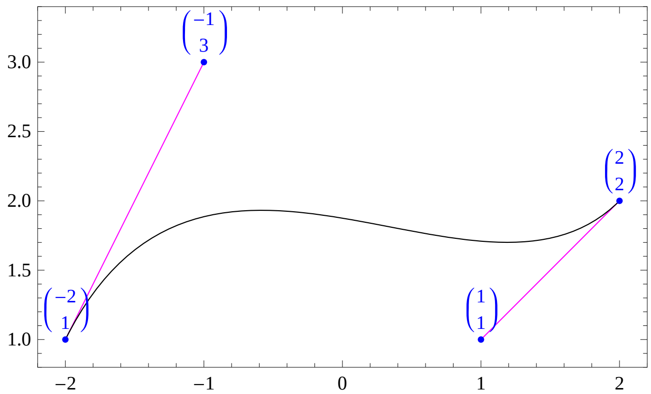
## How SVG and PDF formats draw curves: Cubic splines

If you dig through the reference manuals for
Portable Data Format (PDF), http://www.adobe.com/devnet/acrobat/pdfs/PDF32000_2008.pdf ,
or Scalable Vector Graphics (SVG) http://www.w3.org/TR/SVG11/ ,
you'll find [PDF reference manual chapter 8.5.2.2. and SVG reference manual chapter 8.3.6 that a curved path is represented as a sequence of *cubic Bézier splines*.

A cubic Bézier spline is a curve that is described by 4 points $P_1, P_2, P_3, P_4$
where $P_1$ and $P_4$ are the end points, and $P_2$ and $P_3$ are control points that define the tangent directions of the curve at the end points.

Here's an example:

```
Graphics[{Magenta, Line /@ Partition[#, 2], Black,
    BezierCurve[#], Blue, Point[#], Text[MatrixForm[#], # + {0, .2}] & /@ #},
   Frame → True, PlotRangePadding → 0.2] &[{{-2, 1}, {-1, 3}, {1, 1}, {2, 2}}]
```



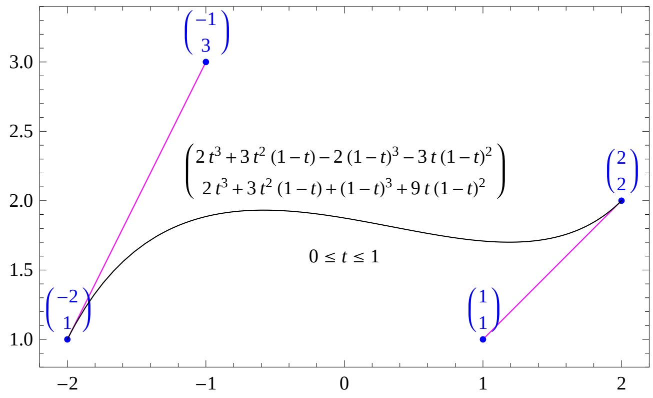

The curve is defined by an equation

```
TraditionalForm[bezier3 = P[1] s^3 + 3 P[2] s^2 t + 3 P[3] s t^2 + P[4] t^3 /. s → 1 - t]
```

$$P(4)\, t^3 + 3\, P(3)\, t^2\, (1-t) + P(1)\, (1-t)^3 + 3\, P(2)\, t\, (1-t)^2$$

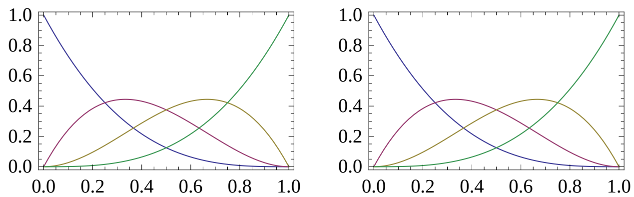
where $0 \leq t \leq 1$.

```
Show[Graphics[{Magenta, Line /@ Partition[#, 2], Black, Text[0 ≤ t ≤ 1, {0, 1.6}],
     Text[MatrixForm[bezier3 /. MapThread[Rule, {Array[P, 4], #}]], {0, 2.2}], Blue,
     Point[#], Text[MatrixForm[#], # + {0, .2}] & /@ #}, Frame → True, PlotRangePadding → 0.2],
   ParametricPlot[bezier3 /. MapThread[Rule, {Array[P, 4], #}], {t, 0, 1},
    PlotStyle → Black]] &[{{-2, 1}, {-1, 3}, {1, 1}, {2, 2}}]
```




## Cubic Bézier spline as sum of Bernstein polynomials

```
Show[GraphicsArray[
  {Plot[Evaluate[Table[BernsteinBasis[3, n, t], {n, 0, 3}]], {t, 0, 1}, Frame → True],
   Plot[Evaluate[Table[Binomial[3, n] (1 - t)^(3-n) t^n, {n, 0, 3}]], {t, 0, 1}, Frame → True]}]]
```




```
Sum[BernsteinBasis[3, n, t] P[n + 1], {n, 0, 3}]
```

```
BernsteinBasis[3, 0, t] P[1] + BernsteinBasis[3, 1, t] P[2] +
 BernsteinBasis[3, 2, t] P[3] + BernsteinBasis[3, 3, t] P[4]
```

## Recurrence relation

```
recurrence = {β[i_Integer, j_Integer /; j > 0] → β[i, j - 1] (1 - t0) + β[i + 1, j - 1] t0,
  β[i_Integer, 0] → P[i + 1]}
```

```
{β[i_Integer, j_Integer /; j > 0] → (1 - t0) β[i, -1 + j] + t0 β[1 + i, -1 + j],
 β[i_Integer, 0] → P[1 + i]}
```

```
β[0, 3] /. recurrence
% /. recurrence
% /. recurrence
% /. recurrence
```

```
(1 - t0) β[0, 2] + t0 β[1, 2]

(1 - t0) ((1 - t0) β[0, 1] + t0 β[1, 1]) + t0 ((1 - t0) β[1, 1] + t0 β[2, 1])

(1 - t0) ((1 - t0) ((1 - t0) β[0, 0] + t0 β[1, 0]) + t0 ((1 - t0) β[1, 0] + t0 β[2, 0])) +
 t0 ((1 - t0) ((1 - t0) β[1, 0] + t0 β[2, 0]) + t0 ((1 - t0) β[2, 0] + t0 β[3, 0]))

(1 - t0) ((1 - t0) ((1 - t0) P[1] + t0 P[2]) + t0 ((1 - t0) P[2] + t0 P[3])) +
 t0 ((1 - t0) ((1 - t0) P[2] + t0 P[3]) + t0 ((1 - t0) P[3] + t0 P[4]))
```

```
β[0, 3] //. recurrence /. (1 - t0) → s0
Simplify[%]
TraditionalForm[% /. s0 → 1 - t0]
```

```
s0 (s0 (s0 P[1] + t0 P[2]) + t0 (s0 P[2] + t0 P[3])) + t0 (s0 (s0 P[2] + t0 P[3]) + t0 (s0 P[3] + t0 P[4]))
```

$s0^3 \, P[1] + 3\, s0^2\, t0\, P[2] + 3\, s0\, t0^2\, P[3] + t0^3\, P[4]$

$P(4)\, t0^3 + 3\, P(3)\, t0^2\, (1 - t0) + P(1)\, (1 - t0)^3 + 3\, P(2)\, t0\, (1 - t0)^2$

## How cairo Draws Splines

Cairo http://cairographics.org is the graphics library used by inkscape.

Cairo uses the de Casteljau algorithm to split splines in half recursively until an approximation by a straight line differs by less than a defined error measure from the spline curve.

From http://cgit.freedesktop.org/cairo/tree/src/cairo- spline.c
and http://cgit.freedesktop.org/cairo/tree/src/cairo- types- private.h :

### ▪ _cairo_spline_error_squared

This function computes a supremum of the maximal error made when approximating a cubic spline by a straight line

A spline is defined by 4 points $\{A, B, C, D\}$, corresponding to $\{P_1, P_2, P_3, P_4\}$ in the computations above.

The following computation is performed:

$$\Delta = D - A$$
$$\beta = B - A$$
$$\beta = \begin{cases} \beta = B - A & \text{if } \beta \cdot \Delta \leq 0 \\ \beta - \Delta = B - D & \text{if } \beta \cdot \Delta \geq \Delta \cdot \Delta \\ \beta - \frac{\beta \cdot \Delta}{\Delta \cdot \Delta} \Delta & \text{else} \end{cases}$$
$$\epsilon_B = \beta \cdot \beta$$
$$\chi = C - A$$
$$\chi = \begin{cases} \chi = C - A & \text{if } \chi \cdot \Delta \leq 0 \\ \chi - \Delta = C - D & \text{if } \chi \cdot \Delta \geq \Delta \cdot \Delta \\ \chi - \frac{\chi \cdot \Delta}{\Delta \cdot \Delta} \Delta & \text{else} \end{cases}$$
$$\epsilon_C = \chi \cdot \chi$$
$$\epsilon = \max \{\epsilon_B, \epsilon_C\}$$

i.e., the larger of the distances, squared, of the control points $\{B, C\}$ (or $\{P_2, P_3\}$ from the line $\overline{\mathrm{AD}}$ (or $\overline{P_1 P_4}$) is computed.

```
SplineErrorSquared = Function[{a, b, c, d},
  Block[{Δ = d - a, β = b - a, χ = c - a},
   Max[(♯.♯ &[♯ - Which[♯.Δ ≤ 0, 0, ♯.Δ ≥ Δ.Δ, Δ, True, (♯.Δ)/(Δ.Δ) Δ]]) & /@ {β, χ}]
  ]
 ]
```

```
Function[{a, b, c, d}, Block[{Δ = d - a, β = b - a, χ = c - a},
  Max[((♯1.♯1 &)[♯1 - Which[♯1.Δ ≤ 0, 0, ♯1.Δ ≥ Δ.Δ, Δ, True, (♯1.Δ Δ)/(Δ.Δ)]] &) /@ {β, χ}]]]
```

```
SplineErrorSquared[{-2, 1}, {-1, 3}, {1, 1}, {2, 2}]
```

$$\frac{49}{17}$$

## _de_casteljau

The control points of two cubic splines covering the ranges $0 \le t \le \frac{1}{2}$ and $\frac{1}{2} \le t \le 1$ are computed from the arithmetic means of point pairs.

```
lerphalf[☺_, ☹_] := ToExpression[ToString[☺] <> ToString[☹]] → Mean[ToExpression /@ {☺, ☹}]
decasteljaurules = lerphalf @@ ♯ & /@ {{a, b}, {b, c}, {c, d}, {ab, bc}, {bc, cd}, {abbc, bccd}}
```

$$\left\{ab \to \frac{a+b}{2},\ bc \to \frac{b+c}{2},\ cd \to \frac{c+d}{2},\ abbc \to \frac{ab+bc}{2},\ bccd \to \frac{bc+cd}{2},\ abbcbccd \to \frac{abbc+bccd}{2}\right\}$$

```
{{a, ab, abbc, abbcbccd}, {abbcbccd, bccd, cd, d}} //. decasteljaurules
Simplify[%]
```

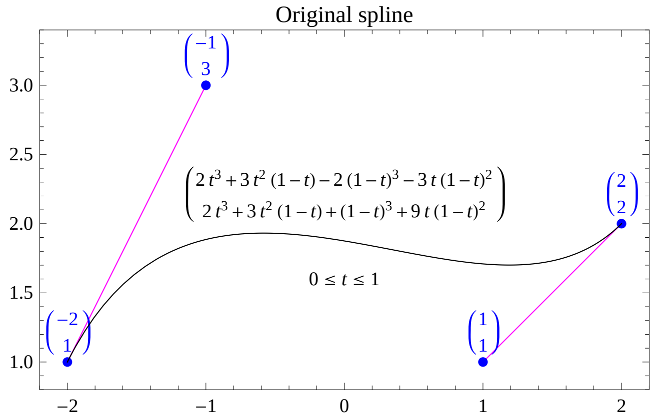
$$\left\{\left\{a, \frac{a+b}{2}, \frac{1}{2}\left(\frac{a+b}{2}+\frac{b+c}{2}\right), \frac{1}{2}\left(\frac{1}{2}\left(\frac{a+b}{2}+\frac{b+c}{2}\right)+\frac{1}{2}\left(\frac{b+c}{2}+\frac{c+d}{2}\right)\right)\right\},\right.$$
$$\left.\left\{\frac{1}{2}\left(\frac{1}{2}\left(\frac{a+b}{2}+\frac{b+c}{2}\right)+\frac{1}{2}\left(\frac{b+c}{2}+\frac{c+d}{2}\right)\right), \frac{1}{2}\left(\frac{b+c}{2}+\frac{c+d}{2}\right), \frac{c+d}{2}, d\right\}\right\}$$

$$\left\{\left\{a, \frac{a+b}{2}, \frac{1}{4}(a+2b+c), \frac{1}{8}(a+3b+3c+d)\right\}, \left\{\frac{1}{8}(a+3b+3c+d), \frac{1}{4}(b+2c+d), \frac{c+d}{2}, d\right\}\right\}$$

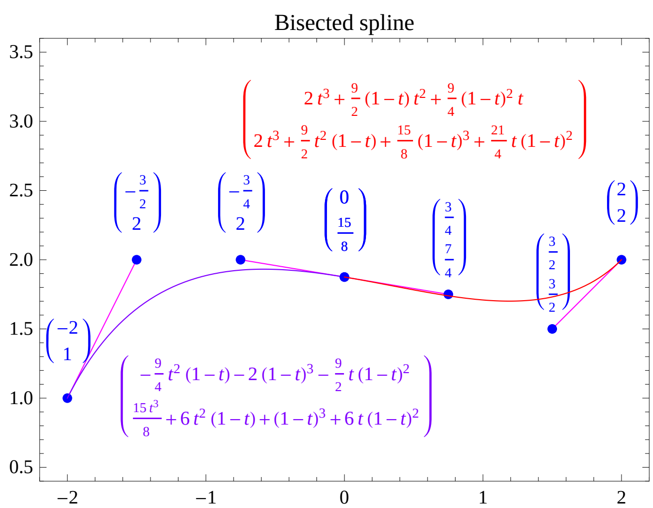
```
deCasteljau = Function[{a, b, c, d},
  Evaluate[{{a, ab, abbc, abbcbccd}, {abbcbccd, bccd, cd, d}} //. decasteljaurules]]
```

$$\text{Function}\left[\{a, b, c, d\}, \left\{\left\{a, \frac{a+b}{2}, \frac{1}{2}\left(\frac{a+b}{2}+\frac{b+c}{2}\right), \frac{1}{2}\left(\frac{1}{2}\left(\frac{a+b}{2}+\frac{b+c}{2}\right)+\frac{1}{2}\left(\frac{b+c}{2}+\frac{c+d}{2}\right)\right)\right\},\right.\right.$$
$$\left.\left.\left\{\frac{1}{2}\left(\frac{1}{2}\left(\frac{a+b}{2}+\frac{b+c}{2}\right)+\frac{1}{2}\left(\frac{b+c}{2}+\frac{c+d}{2}\right)\right), \frac{1}{2}\left(\frac{b+c}{2}+\frac{c+d}{2}\right), \frac{c+d}{2}, d\right\}\right\}\right]$$

```
Show[Graphics[{Magenta, Line /@ Partition[#, 2], Black, Text[0 ≤ t ≤ 1, {0, 1.6}],
     Text[MatrixForm[bezier3 /. MapThread[Rule, {Array[P, 4], #}]], {0, 2.2}],
     Blue, PointSize[Medium], Point[#], Text[MatrixForm[#], # + {0, .2}] & /@ #},
    Frame → True, PlotLabel → "Original spline", PlotRangePadding → 0.2],
   ParametricPlot[bezier3 /. MapThread[Rule, {Array[P, 4], #}], {t, 0, 1},
    PlotStyle → Black]] &[{{-2, 1}, {-1, 3}, {1, 1}, {2, 2}}]
```

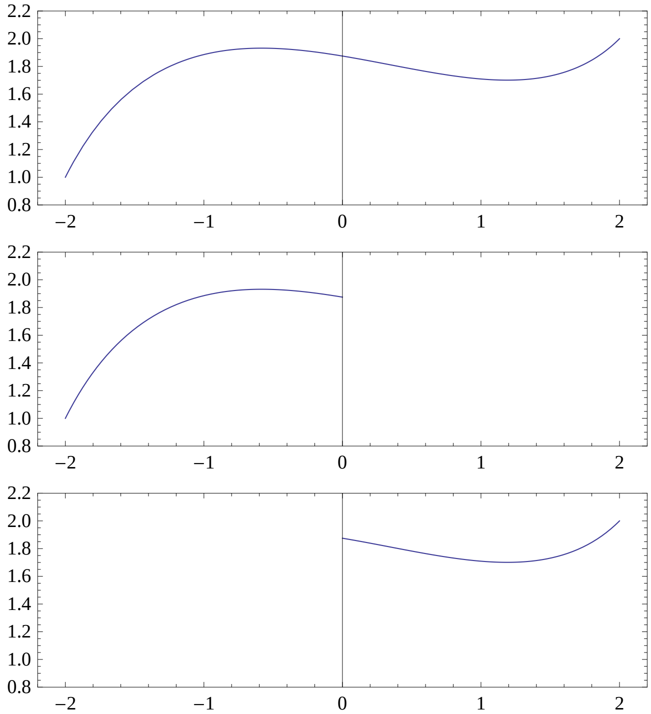



```
Show[MapIndexed[{Graphics[{Magenta, Line /@ Partition[#, 2], Hue[(#2[[1]] + 2)/4],
     Text[MatrixForm[bezier3 /. MapThread[Rule, {Array[P, 4], #}]], {# - 3/2, 2 # - 1} &[#2[[1]]]],
     Blue, PointSize[Medium], Point[#], Text[MatrixForm[#], # + {0, .4}] & /@ #}, Frame → True,
    PlotLabel → "Bisected spline", PlotRangePadding → {0.2, 0.6}], ParametricPlot[
    bezier3 /. MapThread[Rule, {Array[P, 4], #}], {t, 0, 1}, PlotStyle → Hue[(#2[[1]] + 2)/4]]} &,
  deCasteljau[{-2, 1}, {-1, 3}, {1, 1}, {2, 2}]
 ]
]
```

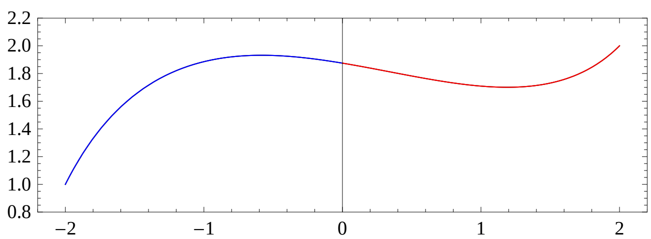


- **Sanity check: Verify that the splines are congruent**

```
TableForm[ParametricPlot[Evaluate[bezier3 /. MapThread[Rule, {Array[P, 4], #}]],
    {t, 0, 1}, Frame → True, PlotRange → {{-2.2, 2.2}, {0.8, 2.2}}] & /@
  (Prepend[deCasteljau @@ #, #] &[{{-2, 1}, {-1, 3}, {1, 1}, {2, 2}}])]
```




```
ParametricPlot[Evaluate[bezier3 /. MapThread[Rule, {Array[P, 4], #}] & /@
   (Prepend[deCasteljau @@ #, #] &[{{-2, 1}, {-1, 3}, {1, 1}, {2, 2}}])], {t, 0, 1},
 Frame → True, PlotRange → {{-2.2, 2.2}, {0.8, 2.2}}, PlotStyle → {Black, Blue, Red}]
```




- **Suprema of errors when approximated by straight line**

```
N[√(SplineErrorSquared[{-2, 1}, {-1, 3}, {1, 1}, {2, 2}])]
N[√(SplineErrorSquared @@ #) & /@ deCasteljau[{-2, 1}, {-1, 3}, {1, 1}, {2, 2}]]
1.69775
{0.715748, 0.467837}
```

## _cairo_spline_decompose
## _cairo_spline_decompose_into

The object **cairo_spline_t** that describes a spline in cairo contains a component **closure** that is filled with a list of points.

If a spline can be approximated by a straight line to within a given tolerance, the first control point is added to the **closure**. Otherwise, the spline is split in two by the de Casteljau algorithm, and the decomposition algorithm is applied to both splines.

```
SplineDecomposeInto = Function[{controlpoints, tolerance},
  If[SplineErrorSquared @@ controlpoints < tolerance,
   Point[First[controlpoints]],
   SplineDecomposeInto[#, tolerance] & /@ deCasteljau @@ controlpoints]
 ]
```

```
Function[{controlpoints, tolerance},
 If[SplineErrorSquared @@ controlpoints < tolerance, Point[First[controlpoints]],
  (SplineDecomposeInto[#1, tolerance] &) /@ deCasteljau @@ controlpoints]]
```

The last control point of the original control point set is added after recursively decomposing the spline.

```
SplineDecompose = Function[{controlpoints, tolerance},
  Flatten[Append[SplineDecomposeInto[controlpoints, tolerance], Point[Last[controlpoints]]]]
 ]
```

```
Function[{controlpoints, tolerance},
 Flatten[Append[SplineDecomposeInto[controlpoints, tolerance], Point[Last[controlpoints]]]]]
```

### Sanity check: Represent spline as set of points and as polyline

```
TableForm[SplineDecompose[{{-2, 1}, {-1, 3}, {1, 1}, {2, 2}}, .02]]
Line[% /. Point → Identity]
```

Point[{-2, 1}]

Point$\left[\left\{-\frac{405}{256}, \frac{807}{512}\right\}\right]$

Point$\left[\left\{-\frac{35}{32}, \frac{119}{64}\right\}\right]$

Point$\left[\left\{0, \frac{15}{8}\right\}\right]$

Point$\left[\left\{\frac{35}{32}, \frac{109}{64}\right\}\right]$

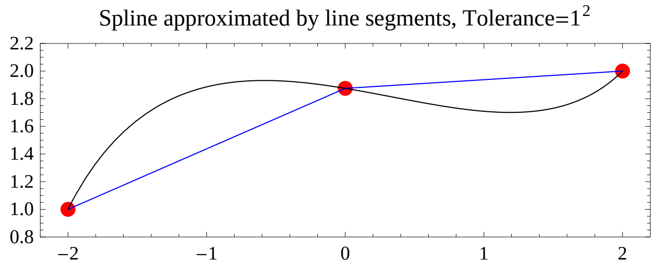
Point$\left[\left\{\frac{405}{256}, \frac{897}{512}\right\}\right]$

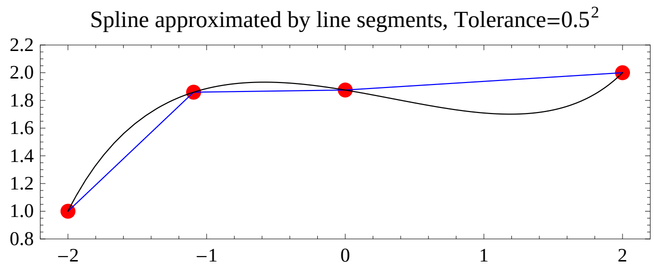
Point[{2, 2}]

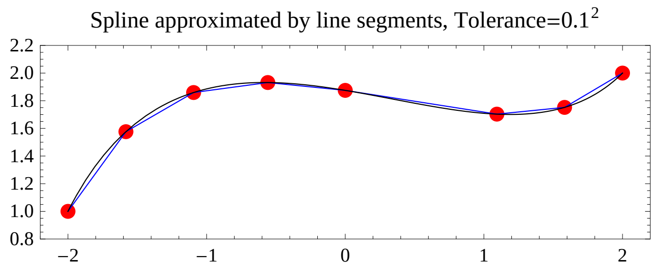
Line$\left[\left\{\{-2, 1\}, \left\{-\frac{405}{256}, \frac{807}{512}\right\}, \left\{-\frac{35}{32}, \frac{119}{64}\right\}, \left\{0, \frac{15}{8}\right\}, \left\{\frac{35}{32}, \frac{109}{64}\right\}, \left\{\frac{405}{256}, \frac{897}{512}\right\}, \{2, 2\}\right\}\right]$

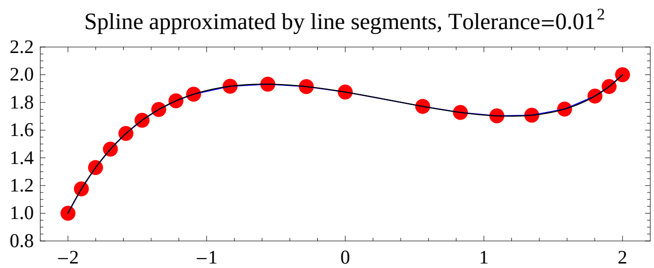
```
splineapprox = Function[{controlpoints, tolerance},
   List[Graphics[{Red, PointSize[Large], #, Blue, Line[# /. Point → Identity]} &[
       SplineDecompose[#, tolerance^2]], Frame → True, PlotLabel →
       "Spline approximated by line segments, Tolerance=" <> ToString[tolerance] <> "²",
      PlotRangePadding → 0.2], ParametricPlot[bezier3 /. MapThread[Rule, {Array[P, 4], #}],
      {t, 0, 1}, PlotStyle → Black]] &[controlpoints]];

Print[Show[#]] & /@ (splineapprox[{{-2, 1}, {-1, 3}, {1, 1}, {2, 2}}, #] & /@ {1, 0.5, 0.1, .01});
```


Spline approximated by line segments, Tolerance=1^2
2.2
2.0
1.8
1.6
1.4
1.2
1.0
0.8
−2
−1
0
1
2


Spline approximated by line segments, Tolerance=0.5^2
2.2
2.0
1.8
1.6
1.4
1.2
1.0
0.8
−2
−1
0
1
2


Spline approximated by line segments, Tolerance=0.1^2
2.2
2.0
1.8
1.6
1.4
1.2
1.0
0.8
−2
−1
0
1
2


Spline approximated by line segments, Tolerance=0.01^2
2.2
2.0
1.8
1.6
1.4
1.2
1.0
0.8
−2
−1
0
1
2