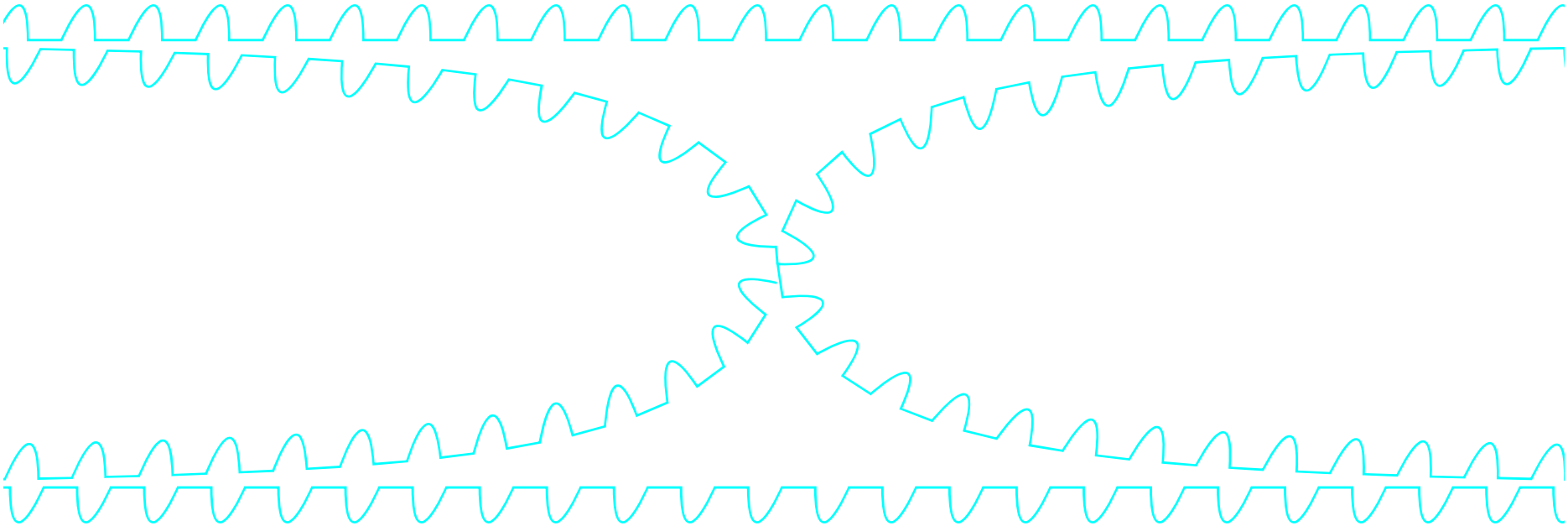

Рисунок 1 - Кривая Каппа

```
import java.awt.Rectangle;
import java.awt.Shape;
import java.awt.geom.AffineTransform;
import java.awt.geom.PathIterator;
import java.awt.geom.Point2D;
import java.awt.geom.Rectangle2D;

public class fooShape implements Shape {

private int param = 0;
private final double delta = 0.001;
private double startAngle = -Math.PI / 2 + delta;
private double endAngle = startAngle + 2 * Math.PI;
private float centerX = 0;
private float centerY = 0;

public fooShape(int param, int centerX, int centerY) {
this.param = param;
this.centerX = centerX;
this.centerY = centerY;
}

@Override
public Rectangle getBounds() {
return null;
}

@Override
public Rectangle2D getBounds2D() {
return null;
}
```

```
@Override
public boolean contains(double x, double y) {
return false;
}

@Override
public boolean contains(Point2D p) {
return false;
}

@Override
public boolean intersects(double x, double y, double w, double h) {
return false;
}

@Override
public boolean intersects(Rectangle2D r) {
return false;
}

@Override
public boolean contains(double x, double y, double w, double h) {
return false;
}

@Override
public boolean contains(Rectangle2D r) {
return false;
}

@Override
public PathIterator getPathIterator(AffineTransform transform) {
```

```
return new PlotIterator(transform);
}

@Override
public PathIterator getPathIterator(AffineTransform transform, double flatness) {
return new PlotIterator(transform, flatness);
}

class PlotIterator implements PathIterator {
AffineTransform transform;
double flatness;
double angle = startAngle;
double step = 10;
boolean done = false;

public PlotIterator(AffineTransform transform, double flatness) {
this.transform = transform;
this.flatness = flatness;
}

public PlotIterator(AffineTransform transform) {
this.transform = transform;
this.flatness = 0;
}

@Override
public int getWindingRule() {
return WIND_NON_ZERO;
}

@Override
public boolean isDone() {
```

```
return done;
}

@Override
public void next() {
if (done)
return;

if (flatness == 0)
step = 0.05;
else
step = flatness;

angle += step;

if (angle >= endAngle)
done = true;
}

@Override
public int currentSegment(float[] coords) {

coords[0] = (float) (param * Math.pow(Math.cos(angle), 2) / Math.sin(angle)) + centerX;
coords[1] = -(float) (param * Math.cos(angle)) + centerY;

if (angle >= endAngle)
done = true;
if (angle == startAngle)
return SEG_MOVETO;
else
return SEG_LINETO;
}
```

```
@Override
public int currentSegment(double[] coords) {

coords[0] = (float) (param * Math.pow(Math.cos(angle), 2) / Math.sin(angle)) + centerX;
coords[1] = -(float) (param * Math.cos(angle)) + centerY;

if (angle >= endAngle)
done = true;
if (angle == startAngle)
return SEG_MOVETO;
else
return SEG_LINETO;
}
}

}
```