# Anisotropic Bump Mapping: Deriving the Full Local Frame

Goal: given a height field and its local gradients, compute the bumped tangent directions U' and V' together with the bumped normal N'.

## 1. Height-field gradients

Assume the finite differences are

```
hx = h(x+1,y) - h(x,y)
hy = h(x,y+1) - h(x,y)
```

The second expression is written with y+1; using x+1 in both gradients would make hx and hy identical.

## 2. Bumped normal

A local height surface can be written as S(x,y) = (x, y, h(x,y)). Its two differential tangent directions are

```
Sx = (1, 0, hx) Sy = (0, 1, hy)
```

Their cross product is

```
Sx x Sy = (-hx, -hy, 1)
```

Therefore the normalized bumped normal is

```
N' = c (-hx, -hy, 1), c = 1 / sqrt(1 + hx^2 + hy^2)
```

## 3. Why U' and V' are not unique

The normal alone does not determine the anisotropic frame. After tilting N into N', the tangent frame can still be rotated by any angle around N'. That twist changes the appearance of an anisotropic BRDF.

A natural convention is to apply the smallest rotation that maps N = (0,0,1) to N'. This tilts the original frame without adding an arbitrary twist.

## 4. Minimal-rotation solution

Let

```
t = hx^2 + hy^2, c = 1 / sqrt(1+t)
```

Then the rotated tangent vectors are

```
U' = ( 1 - (1-c)hx^2/t, -(1-c)hx hy/t, c hx )
```

```
V' = ( -(1-c)hx hy/t, 1 - (1-c)hy^2/t, c hy )
```

Equivalent expanded forms are

```
U' = ( (c hx^2 + hy^2)/t, (c-1)hx hy/t, c hx )
V' = ( (c-1)hx hy/t, (hx^2 + c hy^2)/t, c hy )
```

## 5. Frame properties

The resulting frame is right-handed and orthonormal:

```
|U'| = |V'| = |N'| = 1
U' dot V' = U' dot N' = V' dot N' = 0
U' x V' = N'
```

For hx = hy = 0, the formulas reduce continuously to the original frame U=(1,0,0), V=(0,1,0), N=(0,0,1). In code, handle this case explicitly to avoid dividing by t.

## 6. GLSL implementation

```
void bumpAnisotropicFrame(
    float hx,
    float hy,
    out vec3 U,
    out vec3 V,
    out vec3 N)
{
    float t = hx * hx + hy * hy;

    if (t < 1e-6) {
        U = vec3(1.0, 0.0, 0.0);
        V = vec3(0.0, 1.0, 0.0);
        N = vec3(0.0, 0.0, 1.0);
        return;
    }

    float c = inversesqrt(1.0 + t);
    float k = (1.0 - c) / t;

    U = vec3(
        1.0 - k * hx * hx,
        -k * hx * hy,
        c * hx
    );

    V = vec3(
        -k * hx * hy,
        1.0 - k * hy * hy,
        c * hy
    );

    N = vec3(
        -c * hx,
        -c * hy,
        c
    );
}
```

## 7. Form matching the earlier P/A construction

With

```
P = (-hx, -hy, -t) A = (-hy, hx, 0)
```

the same frame can be written as

```
U' = (-c hx/t) P + (-hy/t) A
V' = (-c hy/t) P + ( hx/t) A
```

```
float t = hx * hx + hy * hy;

if (t < 1e-6) {
    U = vec3(1.0, 0.0, 0.0);
    V = vec3(0.0, 1.0, 0.0);
    N = vec3(0.0, 0.0, 1.0);
    return;
}

float c = inversesqrt(1.0 + t);
vec3 P = vec3(-hx, -hy, -t);
vec3 A = vec3(-hy,  hx,  0.0);

U = (-c * hx / t) * P + (-hy / t) * A;
V = (-c * hy / t) * P + ( hx / t) * A;
```

```
N = vec3(-c * hx, -c * hy, c);
```

## 8. Practical interpretation

For isotropic shading, only N' matters. For anisotropic shading, U' and V' define the local material directions, such as the brushing direction of metal or the fiber direction of silk. The minimal-rotation construction preserves the original anisotropy orientation as much as possible while following the bumped surface.

## Final result

Use the normalized N' above and rotate U and V by the same minimal rotation. The formulas produce a complete orthonormal TBN frame suitable for anisotropic BRDF evaluation.