Ex $\quad V = \dfrac{\vec{p}\cdot\vec{r}}{r^3}$ could be approached

three ways.

I. All Cartesian

II. All Spherical

III. A mix using the product rule

III. Product Rule

$$\nabla(fg) = g\nabla f + f\nabla g$$

Let $f = \frac{1}{r^3}$ $\quad g = \vec{p}\cdot\vec{r}$

In Cartesian, $\vec{p} = (p_x, p_y, p_z)$

$$\vec{r} = (x, y, z)$$

$$\vec{p}\cdot\vec{r} = p_x x + p_y y + p_z z$$

$$\nabla(\vec{p}\cdot\vec{r}) = (p_x, p_y, p_z) = \vec{p}$$

In spherical,

$$\nabla\frac{1}{r^3} = \frac{\partial}{\partial r}\left(\frac{1}{r^3}\right)\hat{r} = -3\frac{1}{r^4}\hat{r}$$

$$\Rightarrow \quad \nabla\left(\frac{\vec{p}\cdot\vec{r}}{r^3}\right) = \vec{p}\cdot\vec{r}\left(-\frac{3}{r^4}\hat{r}\right) + \frac{\vec{p}}{r^3}$$

$$= \frac{1}{r^3}\left(\vec{p} - 3(\vec{p}\cdot\hat{r})\hat{r}\right)$$

Note, $\vec{p}$ does not imply a representation

<u>Method II</u>   <u>All Cartesian</u>

$$\vec{p} = (p_x, p_y, p_z) \qquad \vec{r} = (x, y, z)$$

$$r = \sqrt{x^2+y^2+z^2}$$

$$V = \frac{p_x x + p_y y + p_z z}{\sqrt{x^2+y^2+z^2}}$$

Take grad in Cartesian and get a mess, but a correct mess.

<u>Method III</u>   All spherical

$$\vec{r} = r\hat{r} \qquad \vec{p} = ?$$

$$\vec{p} = p_r\hat{r} + p_\theta\hat{\theta} + p_\phi\hat{\phi}$$

$$\vec{p}\cdot\vec{r} = r p_r$$

$$V = \frac{r\,p_r}{r^3} = \frac{p_r}{r^2}$$

<u>Problem</u> p_r is not constant, it depends on θ, ϕ.
We need to express it in terms of constant p_x, p_y, p_z

$$\vec{p} = p_x \hat{x} + p_y \hat{y} + p_z \hat{z} = p_r \hat{r} + p_\theta \hat{\theta} + p_\phi \hat{\phi}$$

$$\vec{p} \cdot \hat{r} = p_x \hat{x} \cdot \hat{r} + p_y \hat{y} \cdot \hat{r} + p_z \hat{z} \cdot \hat{r} = p_r$$

$$\left.\begin{aligned} \hat{x} \cdot \hat{r} &= \sin\theta \cos\phi \\ \hat{y} \cdot \hat{r} &= \sin\theta \sin\phi \\ \hat{z} \cdot \hat{r} &= \cos\theta \end{aligned}\right]$$

Transformation equations back cover

$$p_r = p_x \sin\theta \cos\phi + p_y \sin\theta \sin\phi + p_z \cos\theta$$

$$V = \frac{p_x \sin\theta \cos\phi + p_y \sin\theta \sin\phi + p_z \cos\theta}{r^2}$$

Take grad using Maple (or by hand).

```
> with(VectorCalculus);
Warning, the assigned names `<,>` and `<|>` now have a global binding
Warning, these protected names have been redefined and unprotected:
`*`, `+`, `-`, `.`, D, Vector, diff, int, limit, series
```

$$[\&x, *, +, -, ., <,>, <|>, AddCoordinates, ArcLength, BasisFormat, Binormal, CrossProd, CrossProduct, Curl, Curvature, \mathrm{D}, Del, DirectionalDiff, Divergence, DotProd, DotProduct, Flux, GetCoordinateParameters, GetCoordinates, Gradient, Hessian, Jacobian, Laplacian, LineInt, MapToBasis, Nabla, Norm, Normalize, PathInt, PrincipalNormal, RadiusOfCurvature, ScalarPotential, SetCoordinateParameters, SetCoordinates, SurfaceInt, TNBFrame, Tangent, TangentLine, TangentPlane, TangentVector, Torsion, Vector, VectorField, VectorPotential, Wronskian, diff, evalVF, int, limit, series] \tag{1}$$

```
> SetCoordinates( 'spherical'[r, φ, θ] );
```

$$spherical_{r,\,\phi,\,\theta} \tag{2}$$

```
> V := (px · sin(theta) · cos(phi) + py · sin(theta) · sin(phi) + pz · cos(theta)) / r^2;
```

$$V := \frac{px \sin(\theta) \cos(\phi) + py \sin(\theta) \sin(\phi) + pz \cos(\theta)}{r^2} \tag{3}$$

```
> Gradient(V);
```

$$-\frac{2\,(px \sin(\theta)\cos(\phi) + py \sin(\theta)\sin(\phi) + pz\cos(\theta))}{r^3}\,\bar{e}_r + \left(\frac{-px\sin(\theta)\sin(\phi) + py\sin(\theta)\cos(\phi)}{r^3}\right)\bar{e}_\phi + \left(\frac{px\cos(\theta)\cos(\phi) + py\cos(\theta)\sin(\phi) - pz\sin(\theta)}{r^3\sin(\phi)}\right)\bar{e}_\theta \tag{4}$$

```
>
```

Cartesian

> *assume*(*px*, *real*);

> *assume*(*py*, *real*);

> *assume*(*pz*, *real*);

> *with*(*VectorCalculus*);

```
Warning, the assigned names `<,>` and `<|>` now have a global
binding
Warning, these protected names have been redefined and unprotected:
`*`, `+`, `-`, `.`, D, Vector, diff, int, limit, series
```

[&x, *, +, -, ., <,>, <|>, *AddCoordinates*, *ArcLength*, *BasisFormat*, *Binormal*, *CrossProd*, *CrossProduct*, *Curl*, *Curvature*, D, *Del*, *DirectionalDiff*, *Divergence*, *DotProd*, *DotProduct*, *Flux*, *GetCoordinateParameters*, *GetCoordinates*, *Gradient*, *Hessian*, *Jacobian*, *Laplacian*, *LineInt*, *MapToBasis*, *Nabla*, *Norm*, *Normalize*, *PathInt*, *PrincipalNormal*, *RadiusOfCurvature*, *ScalarPotential*, *SetCoordinateParameters*, *SetCoordinates*, *SurfaceInt*, *TNBFrame*, *Tangent*, *TangentLine*, *TangentPlane*, *TangentVector*, *Torsion*, *Vector*, *VectorField*, *VectorPotential*, *Wronskian*, *diff*, *evalVF*, *int*, *limit*, *series*] (1)

> $p := < px,\ py,\ px >;$

$$p := (px\sim)e_x + (py\sim)e_y + (px\sim)e_z \tag{2}$$

> $r := < x, y, z >;$

$$r := (x)e_x + (y)e_y + (z)e_z \tag{3}$$

> $p.r;$

$$px\sim x + py\sim y + px\sim z \tag{4}$$

> $V := \dfrac{p.r}{Norm(r)^3};$

$$V := \frac{px\sim x + py\sim y + px\sim z}{\left(x^2+y^2+z^2\right)^{(3/2)}} \tag{5}$$

> *SetCoordinates*(*cartesian*[*x*, *y*, *z*]);

$$cartesian_{x,\,y,\,z} \tag{6}$$

> *Gradient*(*V*);

$$\left(\frac{px\sim}{\left(x^2+y^2+z^2\right)^{(3/2)}} - \frac{3\ (px\sim x + py\sim y + px\sim z)\ x}{\left(x^2+y^2+z^2\right)^{(5/2)}}\right)\bar{e}_x + \left(\frac{py\sim}{\left(x^2+y^2+z^2\right)^{(3/2)}} - \frac{3\ (px\sim x + py\sim y + px\sim z)\ y}{\left(x^2+y^2+z^2\right)^{(5/2)}}\right)\bar{e}_y + \left(\frac{px\sim}{\left(x^2+y^2+z^2\right)^{(3/2)}} - \frac{3\ (px\sim x + py\sim y + px\sim z)\ z}{\left(x^2+y^2+z^2\right)^{(5/2)}}\right)\bar{e}_z \tag{7}$$

> *simplify*(%);

$$\left(\frac{-2\,px\sim x^2 + px\sim y^2 + px\sim z^2 - 3\,x\,py\sim y - 3\,z\,px\sim x}{\left(x^2+y^2+z^2\right)^{(5/2)}}\right)\bar{e}_x + \left(\frac{py\sim x^2 - 2\,py\sim y^2 + py\sim z^2 - 3\,y\,px\sim x - 3\,y\,px\sim z}{\left(x^2+y^2+z^2\right)^{(5/2)}}\right)\bar{e}_y \tag{8}$$

$$- \frac{-px\sim x^2 - px\sim y^2 + 2\,px\sim z^2 + 3\,z\,px\sim x + 3\,z\,py\sim y}{\left(x^2 + y^2 + z^2\right)^{(5/2)}}\,\bar{e}_z$$

>