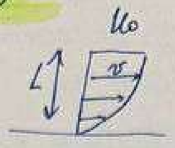
21.02.2024. MA999. HW2 (Blasius solution)



A. Wind flow

1. Define the Reynolds number for our problem. Is the flow laminar or turbulent?

By definition, $Re = \frac{\rho \cdot v \cdot L}{\mu}$, where ρ is density of air, kg/m^3
v is (characteristical) speed of air, m/sec
μ is dynamic viscocity of air, $Pa \cdot sec = \frac{N \cdot sec}{m^2} = \frac{kg}{m \cdot sec}$
L is characteristical length of the profile, in our case, it is the vertical length of our wind profile.

- ρ = air density $= 1.2255 \frac{kg}{m^3}$, under standard conditions and $15°C$.
- $v = 5\ m/c$ – it is typical wind speed in West Midlands. (if we are interested in external wind speed, we can take $10\ m/c$)
- μ = dynamic viscosity $= 18 \cdot 10^{-6} \frac{kg}{m \cdot sec}$
- $L = 40\ cm = 0.4\ m$ (this quantity I took from the typical height of crops and typical height of the beam of the tractor with fertilizers, but later when I computed Blasius solution, it turned out that 30-40 cm well fits with the upper bound of obtained solution! And we can also calculate boundary layer height as $\delta \sim \frac{L}{\sqrt{Re}}$ and obtain the quantity like 30cm)

$\nu = \frac{\mu}{\rho}$ = kinematic viscocity of the air $= \frac{18 \cdot 10^{-6} \cdot \frac{kg}{m \cdot sec}}{1.2255 \frac{kg}{m^3}} = 14.68 \cdot 10^{-6} \cdot \frac{m^2}{sec} \approx 15 \cdot 10^{-6} \cdot \frac{m^2}{sec}$

$\Rightarrow Re = \frac{\rho \cdot v \cdot L}{\mu} = \frac{vL}{\nu} \approx \frac{5 \frac{m}{sec} \cdot 0.4 m}{15 \cdot 10^{-6} \cdot m^2/sec} = \frac{2}{15} \cdot 10^6 = \boxed{1.3 \cdot 10^5}$ – laminar flow

if $v = 10\ m/sec$, then $Re \approx \boxed{2.6 \cdot 10^5}$ – flow that is going from laminar to turbulent.

As it is known, for air if $Re < 2 \cdot 10^5$, then the flow is laminar.
if $Re \in [2 \cdot 10^5; 4 \cdot 10^5]$, then the flow is going from laminar to turbulent.
if $Re > 4 \cdot 10^5$, then the flow is turbulent.

As we will later understand from modelling and agricultural literature, winds $> 5\ m/c$ are considered as bad weather for fertilizing crops, because there are no good enough mechanisms to produce big enough droplets at good speech to prevent the drops from being blown away by the wind. So for our industrial task speed $v = 5\ m/c$ is very reasonable.

Let's see at the size of boundary layer with such Reinold's number. If the length of a field is 100m, then

$\delta \sim \frac{L}{\sqrt{Re}} = \frac{100 m}{\sqrt{1.3 \cdot 10^5}} \approx 0.27 m = 27\ sm.$ – it is well aligned with our assumption that $L = 0.4 m$.

And I also want to emphasize the fact that our flow turned to be laminar is very good – because when one searches information about Blasius solution, it is written that it is in a laminar flow – and our flow is laminar.

(2) Introduce the stream function ψ and re-cast the system of equations and boundary conditions in view of this quantity.

Our initial problem is:

$$\begin{cases} \rho\left(u\cdot\frac{\partial u}{\partial x}+v\cdot\frac{\partial u}{\partial y}\right)=\mu\cdot\frac{\partial^2 u}{\partial y^2} & \leftarrow \text{Navier-Stocks equation in 2D with } \frac{\partial}{\partial t}=0, \text{ and } F_i=0. \\ \frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}=0. & \leftarrow \text{continuity equation} \\ \left.\begin{array}{l} u\big|_{y=0}=0 \\ v\big|_{y=0}=0 \end{array}\right\} & \leftarrow \text{slip condition: at the ground the wind's speed is zero.} \\ u\,(y \text{ big enough})=u_0=\text{const.} \end{cases}$$

here $u(x,y)$ is the x-component of the flow,
$v(x,y)$ is the y-component of the flow

Let's introduce ψ such that $\begin{cases} u=\psi'_y \\ v=-\psi'_x. \end{cases}$ ψ is called a stream function.

then:
$$\begin{cases} u\cdot\frac{\partial u}{\partial x}+v\frac{\partial u}{\partial y}=\nu\frac{\partial^2 u}{\partial y^2} & \Rightarrow \psi'_y\cdot\psi''_{yx}-\psi'_x\cdot\psi''_{yy}=\nu\cdot\psi'''_{yyy} \\ \frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}=0 & \Rightarrow \psi''_{yx}-\psi'_{xy}=0. \\ u\big|_{y=0}=0 & \Rightarrow \psi'_y\big|_{y=0}=0 \\ v\big|_{y=0}=0 & \Rightarrow \psi'_x\big|_{y=0}=0. \\ u\,(x,+\infty)=u_0 & \Rightarrow \psi'_y(x,+\infty)=u_0\equiv\text{const}; \end{cases}$$
in particular, $\psi'_y(x,y)$ should not depend on x, and we will use this when

(3) Consider $x=\beta^a\tilde{x}$; $y=\beta^b\tilde{y}$, $\psi=\beta^c\cdot\tilde{\psi}$, use similarity solutions and obtain Blasius solution: $\begin{cases} f'''+\frac{1}{2}ff''=0 \\ f(0)=f'(0)=0 \\ f'(\eta\to+\infty)\to 1. \end{cases}$

Let's make change of variables:

$\begin{cases} \tilde{x}=\varepsilon^a\cdot x \\ \tilde{y}=\varepsilon^b\cdot y \\ \tilde{\psi}=\varepsilon^c\cdot\psi \end{cases}$ and plug it into our equation $\psi'_y\cdot\psi''_{yx}-\psi'_x\cdot\psi''_{yy}=\nu\cdot\psi'''_{yyy}$

$\Rightarrow$ 1) $\tilde{\psi}'_{\tilde{y}}=\tilde{\psi}'_\psi\cdot\psi'_{\tilde{y}}=\varepsilon^c\cdot\psi'_{\tilde{y}}=\varepsilon^c\cdot\psi'_y\cdot y'_{\tilde{y}}=\varepsilon^c\cdot\psi'_y\cdot\varepsilon^{-b}=\varepsilon^{c-b}\cdot\psi'_y$

2) $\tilde{\psi}''_{\tilde{y}\tilde{x}}=(\tilde{\psi}'_{\tilde{y}})'_{\tilde{x}}=(\varepsilon^{c-b}\cdot\psi'_y)'_{\tilde{x}}=\varepsilon^{c-b}\cdot\psi''_{yx}\cdot x'_{\tilde{x}}=\varepsilon^{c-b}\cdot\psi''_{yx}\cdot\varepsilon^{-a}=\varepsilon^{c-b-a}\cdot\psi''_{yx}$

3) $\tilde{\psi}'_{\tilde{x}}=\tilde{\psi}'_\psi\cdot\psi'_{\tilde{x}}=\varepsilon^c\cdot\psi'_x\cdot x'_{\tilde{x}}=\varepsilon^{c-a}\cdot\psi'_x$

4) $\tilde{\psi}''_{\tilde{y}\tilde{y}}=(\tilde{\psi}'_{\tilde{y}})'_{\tilde{y}}=(\varepsilon^{c-b}\cdot\psi'_y)'_{\tilde{y}}=\varepsilon^{c-b}\cdot\psi''_{yy}\cdot y'_{\tilde{y}}=\varepsilon^{c-2b}\cdot\psi''_{yy}$

5) $\tilde{\psi}'''_{\tilde{y}\tilde{y}\tilde{y}}=(\tilde{\psi}''_{\tilde{y}\tilde{y}})'_{\tilde{y}}=(\varepsilon^{c-2b}\cdot\psi''_{yy})'_{\tilde{y}}=\varepsilon^{c-2b}\cdot\psi'''_{yyy}\cdot y'_{\tilde{y}}=\varepsilon^{c-3b}\cdot\psi'''_{yyy}$

Plug all that into equation: $\underbrace{\psi'_y}_{\varepsilon^{b-c}\cdot\tilde{\psi}'_{\tilde{y}}}\cdot\underbrace{\psi''_{yx}}_{\varepsilon^{a+b-c}\cdot\tilde{\psi}''_{\tilde{y}\tilde{x}}}-\underbrace{\psi'_x}_{\varepsilon^{a-c}\cdot\tilde{\psi}'_{\tilde{x}}}\cdot\underbrace{\psi''_{yy}}_{\varepsilon^{2b-c}\cdot\tilde{\psi}''_{\tilde{y}\tilde{y}}}=\nu\cdot\underbrace{\psi'''_{yyy}}_{\varepsilon^{3b-c}\cdot\tilde{\psi}'''_{\tilde{y}\tilde{y}\tilde{y}}}$

$\Rightarrow \varepsilon^{a+2b-2c}\cdot\tilde{\psi}'_{\tilde y} - \varepsilon^{a+2b-2c}\cdot\tilde{\psi}'_{\tilde x}\cdot\tilde{\psi}''_{\tilde y\tilde y} = \nu\cdot\varepsilon^{3b-c}\cdot\tilde{\psi}'''_{\tilde y\tilde y\tilde y}$ (2)

And in order for the equation to be preserved in form,
we should have $a+2b-2c = 3b-c$

$\Rightarrow a = b+c$

And another constraint on a, b, c we obtain from boundary condition, but later.

Now let's make sure that expressions $\psi\cdot x^{-\frac{c}{a}}$ and $y\cdot x^{-\frac{b}{a}}$ are both preserved in form under this change of variables:

- $\tilde\psi\cdot\tilde x^{-\frac{c}{a}} = (\varepsilon^c\cdot\psi)\cdot(\varepsilon^a\cdot x)^{-\frac{c}{a}} = \psi\cdot x^{-\frac{c}{a}}$
- $\tilde y\cdot\tilde x^{-\frac{b}{a}} = \varepsilon^b y\cdot(\varepsilon^a\cdot x)^{-\frac{b}{a}} = y\cdot x^{-\frac{b}{a}}$

$\Rightarrow$ let's look for a solution in the form: $\psi\cdot x^{-\frac{c}{a}} = f(y\cdot x^{-\frac{b}{a}})$, where $a = b+c$.

$\Rightarrow \psi = x^{\frac{c}{a}}\cdot f(y\cdot x^{-\frac{b}{a}}) = x^{\frac{c}{b+c}}\cdot f(y\cdot x^{-\frac{b}{b+c}})$; and denote $\eta := y\cdot x^{-\frac{b}{b+c}}$

Now look at three boundary conditions:

- $\psi'_y|_{y=0} = 0$: $\psi'_y = x^{\frac{c}{b+c}}\cdot f'_\eta\cdot x^{-\frac{b}{b+c}} = x^{\frac{c-b}{b+c}}\cdot f'_\eta\Big|_{\eta=0} \overset{?}{=} 0$.
- $\psi'_x|_{y=0} = 0$: $\psi'_x = \frac{c}{b+c}\cdot x^{(\frac{c}{b+c}-1)}\cdot f(\eta)\Big|_{y=0} + \underbrace{x^{\frac{c}{b+c}}\cdot f'_\eta\cdot \overset{0}{y}\cdot\left(-\frac{b}{b+c}\right)\cdot x^{\left(\frac{b}{b+c}-1\right)}}_{=0 \text{ at } y=0}\Big|_{y=0} \overset{?}{=} 0$.
- $\psi'_y(x, +\infty) \overset{?}{=} u_0$: $\psi'_y = x^{\frac{c-b}{b+c}}\cdot f'_\eta \xrightarrow{?} u_0$ as $y\to+\infty$, $\forall x$

This should not depend on $x \Rightarrow b = c$ $\Rightarrow \begin{cases} \psi'_y|_{y=0} \overset{?}{=} 0 \text{ gives } f'_\eta|_{\eta=0} = 0. \\ \psi'_x|_{y=0} \overset{?}{=} 0 \text{ gives } f(\eta)|_{\eta=0} = 0. \end{cases}$

$\Rightarrow$ we have $\begin{cases} a = b+c \\ b = c \end{cases}$

$\Rightarrow \psi = x^{\frac{1}{2}}\cdot f(y\cdot x^{-1/2})\cdot \tilde C$ ← some constant

Let's denote $\delta(x) = \sqrt{\frac{\nu x}{u_0}}$; $\eta := \frac{y}{\delta(x)}$ — these coordinates will be very convenient.

$\Rightarrow \psi = u_0\cdot\delta(x)\cdot f(\eta)$ with $f'_\eta|_{\eta=0} = 0$, $f|_{\eta=0} = 0$; and $\psi'_y = u_0\cdot\delta(x)\cdot f'_\eta\cdot\frac{1}{\delta(x)} \overset{?}{\to} u_0$ gives $f'(+\infty) = 1$.

Now let's plug this ψ into our equation on ψ (that came from initial Navier-Stokes equation after change of variables):

$$\boxed{\psi'_y\cdot\psi''_{yx} - \psi'_x\cdot\psi''_{yy} = \nu\cdot\psi'''_{yyy}}$$

$\Rightarrow \psi'_y = u_0\cdot\delta\cdot f'_\eta\cdot\frac{1}{\delta} = u_0 f'_\eta$

$\psi''_{yx} = (\psi'_y)'_x = (u_0 f'_\eta)'_x = u_0\cdot f''_{\eta\eta}\cdot\eta'_x = u_0\cdot f''_{\eta\eta}\cdot\left(-\frac{y}{\delta^2(x)}\cdot\delta'_x\right)$

$\psi'_x = u_0\cdot\delta'_x\cdot f + u_0\cdot\delta\cdot f'_\eta\cdot\eta'_x$

$\psi''_{yy} = (\psi'_y)'_y = (u_0 f'_\eta)'_y = u_0\cdot f''_{\eta\eta}\cdot\frac{1}{\delta}$

$\psi'''_{yyy} = (\psi''_{yy})'_y = u_0\cdot f'''_{\eta\eta\eta}\cdot\frac{1}{\delta^2}$

$\Rightarrow \psi'_y \cdot \psi''_{yx} - \psi'_x \cdot \psi''_{yy} = \nu \cdot \psi'''_{yyy}$ transforms into:

$$\cancel{(u_0 \cdot f'_\eta)(u_0 \cdot f''_{\eta\eta} \cdot \eta'_x)} - (u_0 \cdot \delta'_x \cdot f + \cancel{u_0 \cdot \delta \cdot f'_\eta \cdot \eta'_x})(u_0 \cdot f''_{\eta\eta} \cdot \tfrac{1}{\delta}) = \nu \cdot u_0 \cdot f'''_{\eta\eta\eta} \cdot \tfrac{1}{\delta^2}$$

$$\Rightarrow \frac{-u_0^2}{\delta} \delta'_x \cdot f f''_{\eta\eta} = \nu \cdot \cancel{u_0} \cdot f'''_{\eta\eta\eta} \cdot \frac{1}{\delta^2}$$

$$\Rightarrow \quad u_0 \delta'_x f \cdot f''_{\eta\eta} + \frac{\nu}{\delta} \cdot f'''_{\eta\eta\eta} = 0.$$

$$\left(\sqrt{\frac{\nu x}{u_0}}\right)'_x$$

$$= \sqrt{\frac{\nu}{u_0}} \frac{1}{2\sqrt{x}}$$

$$= \frac{1}{2} \cdot \frac{\nu}{u_0} \cdot \frac{1}{\delta}.$$

$$\Rightarrow \cancel{u_0}\left(\frac{1}{2} \cdot \frac{\cancel{\nu}}{\cancel{u_0}} \cdot \frac{1}{\delta}\right) \cdot f \cdot f''_{\eta\eta} + \frac{\cancel{\nu}}{\delta} \cdot f'''_{\eta\eta\eta} = 0.$$

$$\Rightarrow \boxed{\frac{1}{2} \cdot f \cdot f''_{\eta\eta} + f'''_{\eta\eta\eta} = 0.}$$

And remember the initial conditions we derived: $f|_{\eta=0}$; $f'_\eta|_{\eta=0} = 0$; $f'_\eta(+\infty) = 1$.

$\Rightarrow$ we obtain:

$$\boxed{\begin{cases} f'''_{\eta\eta\eta} + \frac{1}{2} f f''_{\eta\eta} = 0 \\ f|_{\eta=0} = 0 \\ f'_\eta|_{\eta=0} = 0 \\ f'_\eta(+\infty) = 1. \end{cases}}$$

4. Solve the blasius solution

We first need to do smth with initial condition $f'_\eta(+\infty) = 1$.

~~We~~ Standard initial conditions for 3rd order ODE will be:

$f(0)=0$; $f'(0)=0$; $f''(0)=a$, where a is some number, that will give such a solution that has $f'(+\infty)=1$.

To find this a we can use a shooting method.

find two solutions, one with initial conditions $g(0)=0$; $g'(0)=0$; $g''(0)=a$, and the other with init. conditions $h(0)=0$; $h'(0)=0$ $h''(0)=b$

And then say that since if we take a linear combination of them:

$\Rightarrow f = w \cdot g + (1-w) \cdot h$

$\Rightarrow f'(+\infty) = w \cdot g'(+\infty) + (1-w) \cdot h'(+\infty) \overset{?}{=} 1.$

$$\Rightarrow w(g'(+\infty) - h'(+\infty)) \overset{?}{=} 1 - h'(+\infty)$$

$$\Rightarrow w^* = \frac{1 - h'(+\infty)}{g'(+\infty) - h'(+\infty)} \Rightarrow \text{take } f = w^* g + (1-w^*) \cdot h.$$

But since we are using python, we can numerically integrate our ~~equat~~ ODE, by reducing it to a system in 3D:

$y_1 := y \quad \Rightarrow y_1' = y_2$

$y_2 := y' \quad \Rightarrow y_2' = y_3$

$y_3 := y'' \quad \Rightarrow y_3' = -\frac{1}{2} \cdot y \cdot y'' = -\frac{1}{2} \cdot y_1 \cdot y_3$

$$\Rightarrow \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix}' = \begin{pmatrix} y_2 \\ y_3 \\ -\frac{1}{2} y_1 y_3 \end{pmatrix} \rightsquigarrow \text{plug into odeint and get } y, y', y''(a), \text{ where } a \text{ is a parameter.}$$

And now we can use scipy.optimize.root $(y'(+\infty)-1;\ [1.0])$ to find such a, that will give $y'(a)\big|_{\eta=+\infty} - 1 \overset{?}{=} 0$, that is, $y'(+\infty)=1$.

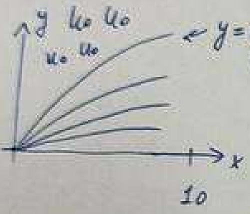
And we will get $a = 0.33205736$.

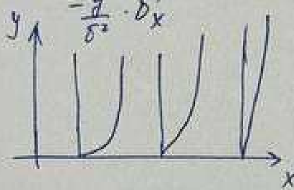
Then, when we get f, f', f'' for all etas (in truth, we got f, f', f'' only on a discretized grid of etas, but since in physics all functions are good and continuous, we set f, f', f'' at other etas to be interpolated f, f', f'' from the grid), we should obtain back $u(x,y)$ and $v(x,y)$:

$$\psi = u_0 \cdot \delta(x) \cdot f(\eta)$$

$$\Rightarrow u = \psi'_y = u_0 \cdot \delta \cdot f'_\eta \cdot \frac{1}{\delta} = \boxed{u_0 \cdot f'_\eta}$$

$$v = -\psi'_x = -u_0 \cdot (\delta'_x \cdot f(\eta) + \delta \cdot f'_\eta \cdot \underbrace{\eta'_x}_{= -\frac{y}{\delta^2} \cdot \delta'_x}) = -u_0 \cdot \delta' (f - f'_\eta \underbrace{\left(\frac{y}{\delta}\right)}_{\eta}) = \boxed{-u_0 \cdot \delta' \cdot (f - f'_\eta \cdot \eta)}$$





← developing Blasius boundary layer (see in jupyter notebook) My picture looks like the picture from Wikipedia.

5. Extend the model to include z-component.

look at 3D Navier-Stokes equations:

$$\rho \cdot \frac{d\bar{v}}{dt} = -\bar{\nabla} p + \mu \nabla^2 \bar{u} + \bar{F}e \cdot \rho$$

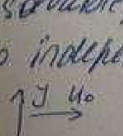
We assume that $\frac{\partial p}{\partial y} = 0$, that means $p = p(x, z)$ – pressure;

- region is sufficiently large and wind is sufficiently strong ~~that~~ to ~~say~~ neglect gravity $\Rightarrow (Fex, Fey, Fez) = 0$.
- The wind front is stable and has reached steady state $\Rightarrow \frac{\partial}{\partial t} = 0$.
- Region is spatially infinite in x and z-directions $\Rightarrow \frac{\partial}{\partial x}, \frac{\partial}{\partial z} \ll \frac{\partial}{\partial y}$

$$\Rightarrow \begin{cases} \rho \cdot \left(u \cdot \frac{\partial u}{\partial x} + v \cdot \frac{\partial u}{\partial y} + w \cdot \frac{\partial u}{\partial z}\right) = -\frac{\partial p}{\partial x} + \mu \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2}\right) \\ \rho \cdot \left(u \cdot \frac{\partial v}{\partial x} + v \cdot \frac{\partial v}{\partial y} + w \cdot \frac{\partial v}{\partial z}\right) = -\frac{\partial p}{\partial y} + \mu \cdot \left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} + \frac{\partial^2 v}{\partial z^2}\right) \\ \rho \left(u \cdot \frac{\partial w}{\partial x} + v \cdot \frac{\partial w}{\partial y} + w \cdot \frac{\partial w}{\partial z}\right) = -\frac{\partial p}{\partial z} + \mu \left(\frac{\partial^2 w}{\partial x^2} + \frac{\partial^2 w}{\partial y^2} + \frac{\partial^2 w}{\partial z^2}\right) \\ \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} = 0. \\ u|_{y=0} = 0 \\ v|_{y=0} = 0 \\ w|_{y=0} = 0 \\ u(y=+\infty) = u_0. \end{cases}$$

And we can also assume that pressure is constant $\Rightarrow \frac{\partial p}{\partial x} = \frac{\partial p}{\partial y} = \frac{\partial p}{\partial z} = 0$.

But to make this 3D system solvable, let's introduce the model that is a multiplication of two independent boundary layers, one in xOy, and the other in yOz



$\Rightarrow$ we have 6 equations solutions and can solve the system.

⑥ Make three suggestions to make the flow more realistic.

1) Clearly make it 3D, for example, think of it as a multiplication of two independent Blasius solutions in XOY and YOZ planes. Because if we review the winds in West Midlands, we will see that the wind is never totally north, or totally east/west/south, and it can change its direction, and for us not to recalculate the whole model (we definitely can put the x-axis in the direction of the wind), we should take into account that the wind is in 3d.

2) Also we should take into consideration the fact that winds depend on time, because winds in the morning and winds in the midday are not the same, and also they have seasonality: winds in summer and in winter differ.

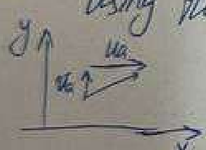
$\Rightarrow$ better model will be $U(x,y,z,t) = U(x,y) \cdot F_2(z) \cdot F_3(t)$, where $F_3(t)$ may be taken sinusoidal, for example.

3) Gravity should also taken into account, because it is negligible only for very-very strong winds, but what if we want to calculate and model our fertilization campaign in a day with almost no wind $\Rightarrow F_{gy} = -mdg \neq 0$.

4) We must take into account the fact that close to earth there is grass $\Rightarrow$ temperature is 1-2°C higher than in the air $\Rightarrow$ there will be an upward force, which should be taken into account.

B. Particle modelling

① Develop a simple model of movement for spherical particles using the Blasius solution flow as hosting flow

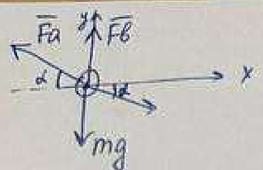


- $(u_a; v_a)$ is the air x- and y-velocities.
  In our case $u_a(x,y)$ and $v_a(x,y)$ are the flow field, found in Ⓐ as a Blasius boundary layer solution
- (u_d, v_d) is the x- and y-velocity of a droplet (of water)
- the droplet is spherical
- the model is 2D for simplicity
- $\begin{cases} u_{rel} = u_d - u_a \\ v_{rel} = v_d - v_a \end{cases}$ is the relative speed of the droplet with respect to air.
- $|U_{rel}| = \sqrt{(u_{rel})^2 + (v_{rel})^2}$ – the absolute value of U_{rel}.
- On a drop acts 3 forces: gravitation force (downwards), buoyancy force (upwards), and aerodynamical force, that is proportional to the square of speed and acts in the direction that is opposite to speed.
- We can write second Newton's law and project it onto 2 axes: x and y.

$m\bar{a} = \bar{F} = F_a + F_g + F_B$

$\Rightarrow m_d \ddot{x} = \bar{F}$; $\dot{x} = (u_d; v_d)$

$$\Rightarrow \begin{cases} \dfrac{du_d}{dt} = (F_{ax} + \overset{=0}{F_{gx}} + \overset{=0}{F_{Bx}})/m_d \\ \dfrac{dv_d}{dt} = (F_{ay} + F_{gy} + F_{By})/m_d. \end{cases}$$



$\vec{mg} + \vec{F_b}$ projected on y axis give:

$$-m_d g + \rho_a \cdot g \cdot V_d = -m_d \cdot g + \rho_a \cdot g \cdot \frac{V_d \cdot \rho_d}{\rho_d} =$$

$$= -m_d \cdot g + \frac{\rho_a}{\rho_d} \cdot g \cdot m_d = -m_d g \left(1 - \frac{\rho_a}{\rho_d}\right).$$

- $|F_a| = \frac{1}{2} \cdot C_d \cdot \rho_a \cdot A_d \cdot |U_{res}|^2$

  where
  - C_d is the drag coefficient, it depends only on the form of the droplet, and for a sphere the drag coefficient is 0.47 (Wikipedia)
  - ρ_a is the density of air, $1.2255 \frac{kg}{m^3}$
  - $A_d = \pi R^2 = \frac{\pi D^2}{4}$ is the frontal area of the drop, where D is the diameter of the drop
  - $U_{res}^2 = (u_d - u_a)^2 + (v_d - v_a)^2$.
- $m_d = \rho_d \cdot V_d = \rho_d \cdot \frac{4}{3} \pi R^3 = \rho_d \frac{4}{3} \pi \frac{D^3}{8}$ is the mass of the droplet; $\rho_a = 1000 \frac{kg}{m^3}$ is the ~~mass~~ density of water.

See that $F_{ax} = -|F_a| \cdot \cos\alpha \cdot \frac{\vec{U}_{res}}{|U_{res}|} = \frac{1}{2} \cdot C_d \cdot \rho_a \frac{\pi D^2}{4} \cos\alpha \frac{|U_{res}|^2 U_{res}}{|U_{res}|}$

unit vector pointing in the direction of U_{res}

$= (U_{res})_{x\,component}$

$$\Rightarrow F_{ax} = -\frac{1}{8} \cdot c_d \cdot \rho_a \cdot \pi D^2 \cdot (u_d - u_a).$$

Analogously, $F_{ay} = -|F_a| \sin\alpha \frac{\vec{U_{res}}}{|U_{res}|} = -\frac{1}{8} c_d \cdot \rho_a \cdot \pi D^2 \cdot (v_d - v_a)$

All Together:

$$\begin{cases} \frac{du_d}{dt} = \frac{F_{ax} + \cancel{F_{gx}}^{0} + \cancel{F_{bx}}^{0}}{m_d} \\ \frac{dv_d}{dt} = \frac{F_{ay} + F_{gy} + F_{by}}{m_d} \end{cases}$$

$$\Rightarrow \begin{cases} \frac{du_d}{dt} = \frac{-\frac{1}{8} c_d \rho_a \cdot \pi D^2 \cdot (u_d - u_a)}{\rho_d \cdot \frac{1}{2 \cdot 3} \pi D^3} = -\frac{3}{4} \frac{c_d \cdot \rho_a \cdot D}{\rho_d} \cdot (u_d - u_a) \\ \frac{dv_d}{dt} = -\frac{3}{4} \frac{c_d \rho_a \cdot D}{\rho_d} \cdot (v_d - v_a) + \frac{\left(-m_d \cdot g \cdot \left(1 - \frac{\rho_a}{\rho_d}\right)\right)}{m_d} = -\frac{3}{4} \frac{c_d \cdot \rho_a \cdot D}{\rho_d} \cdot (v_d - v_a) - g\left(1 - \frac{\rho_a}{\rho_d}\right). \end{cases}$$

And transform this into 4 1d-ODE:

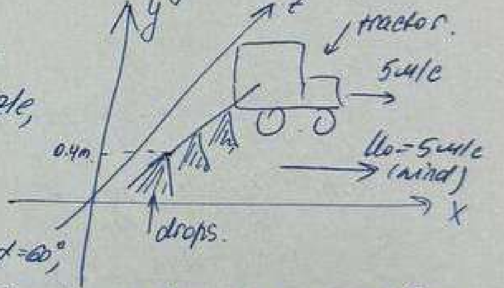
$$\begin{cases} \dot{x} = u_d \\ \dot{u}_d = -\frac{3}{4} \frac{c_d \rho_a D}{\rho_d} (u_d - u_a) \\ \dot{y} = v_d \\ \dot{v}_d = -\frac{3}{4} \frac{c_d \cdot \rho_a \cdot D}{\rho_d} \cdot (v_d - v_a) - g\left(1 - \frac{\rho_a}{\rho_d}\right) \end{cases}$$

+ initial conditions $x(0) = x_0$; $y(0) = y_0$; $\dot{x}(0) = u_0$; $\dot{y}(0) = v_0$

(2.) Justify suitable choices in initial conditions and ejected droplet characteristics

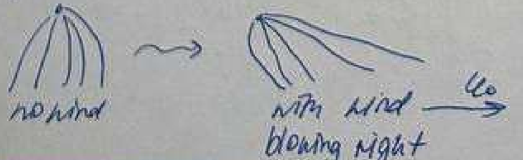
drop size we can find in the brochure about fertilization by Lechler machines. they present different nozzles with different colors, and color corresponds to the size of the droplet that will leave the nozzle (if the pressure in the nozzle will be kept as is written and recommended)
recommended pressure is 3 bar.
Insecticides and fungicides usually are used with drop sizes 100-200 microns, gerbisides - with 100-300 microns. When there is fast wind and/or the drops are evaporating hugely, than greater drop sizes are used, up to 400 microns. 500 microns - are huge drops. And less 100 microns in diameter drops are not used, because they are hugely shifted by the wind and they also evaporate totally untill they reach the plant that we are fertilizing.

About initial conditions: x_0, y_0 is the initial position of the drop - it is clearly the position of the nozzle. Nozzles are located on a beam attached to a tractor, and the heigh of the beam is usually 0.4m - 0.5m. We will use a grid of y_0 from 0 to 0.5m.
As for x, we will use $x=1$, because we assume that the field is spatially uniform, so x_0 are all ~~the~~ having the same properties.
Now let's consider u_0, v_0.



We read from Lechler brochure that depending on pressure inside the nozzle, the drop will leave the nozzle with speed at 20-60 m/sec
and the angles also differs from $\alpha = 40°$ to $\alpha = 60°$, where α is the ~~[illegible]~~ half of the angle of the cone of drops.

And projecting $(u_0; v_0)$ with $\sqrt{|u_0|^2 + |v_0|^2} = 20$ m/sec on x and y axes, we see that $u_0 = 20 \sin\alpha$, and $v_0 = 20 \cos\alpha$.
We will now model trajectories of the drops with different angles, and at different $u_0 = [0.1, 5, 10, 15]$ m/sec to understand, how the wind affects the trajectories inside the cone.
And we will also realize, that the higher the beam is located, the more the wind <u>does</u> affect!



In [29]:
```
import numpy as np
from scipy.integrate import odeint
import scipy
from scipy import interpolate
from itertools import chain
import matplotlib.pyplot as plt
from IPython import display
```

Part A

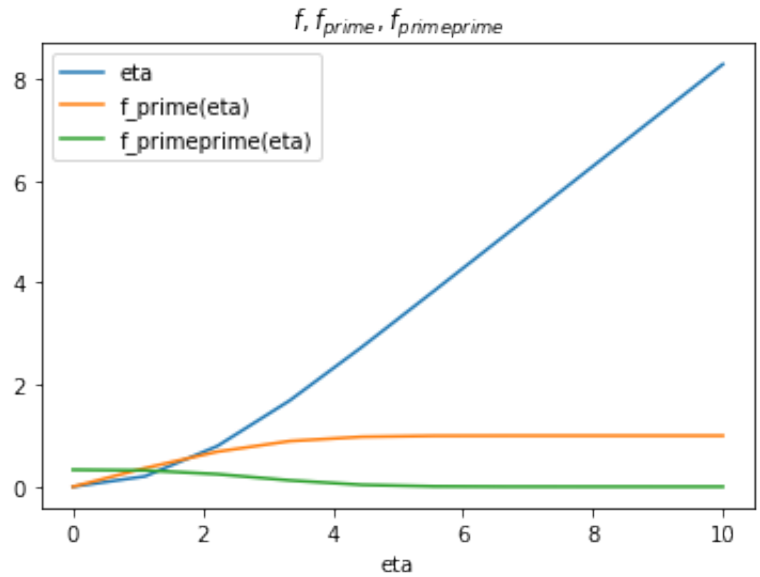
Let's first find the needed value of $a = f''(0)$ that will give us $f'(+\infty) = 1$ -- this a turns out to be 0.33205736;

and then integrate our 3d system of ODE to obtain f, f', f'' on the grid of etas.

In [4]:
```
etas=[x for x in chain(np.linspace(0,100,10000), np.linspace(100,10

def righthandsidefunction(y,eta):
    return [y[1],y[2],-0.5*y[0]*y[2]]

def dydeta_at_infty(a):
    return odeint(righthandsidefunction, [0,0,a], etas)[-1][1] -1.0

a=scipy.optimize.root(dydeta_at_infty,[1.0])
print("Right value of y''(0) is", a.x)

y0=[0,0,a.x] #initial conditions
otv=odeint(righthandsidefunction, y0,etas)
f_of_etas_discretized=list(map(lambda x: x[0],otv))
fprime_of_etas_discretized=list(map(lambda x: x[1],otv))
fprimeprime_of_etas_discretized=list(map(lambda x: x[2],otv))

#U0=5
nu=15*1e-6

def f(eta):
    return interpolate.interp1d(etas, f_of_etas_discretized)([eta])
def fprime(eta):
    return interpolate.interp1d(etas, fprime_of_etas_discretized)([
def fprimeprime(eta):
    return interpolate.interp1d(etas, fprimeprime_of_etas_discretiz
def delta(x,U0):
    return np.sqrt(nu*x/U0)
def deltaprime(x,U0):
    return np.sqrt(nu/(U0*x))/2
def u(x,y,U0):
    eta=y/delta(x,U0)
    return U0*fprime(eta)
def v(x,y,U0):
```

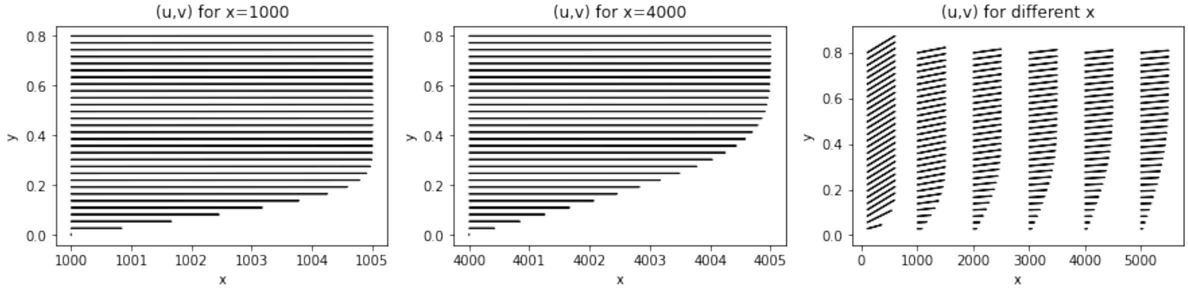
```
    eta=y/delta(x,U0)
    return -U0*deltaprime(x,U0)*(f(eta)-fprime(eta)*eta)

def u_scaled(eta):
    return fprime(eta)
def v_scaled(eta,U0):
    return -deltaprime(x,U0)*(f(eta)-fprime(eta)*eta)


etalims=np.linspace(0,10,10)


plt.plot([eta for eta in etalims],[f(eta) for eta in etalims],label
plt.plot([eta for eta in etalims],[fprime(eta) for eta in etalims],
plt.plot([eta for eta in etalims],[fprimeprime(eta) for eta in etal
plt.xlabel('eta')
plt.title('$f,f_{prime},f_{primeprime}$')
plt.legend()
plt.show()
```

```
Right value of y''(0) is [0.33205736]
```



Now let's draw obtained wind velocities for a fixed $x = 1000$ and $x = 4000$, fixed $U0 = 5$ and different $y \in np.linspace(0, 0.8, 30)$; and then draw velocities for some other x;

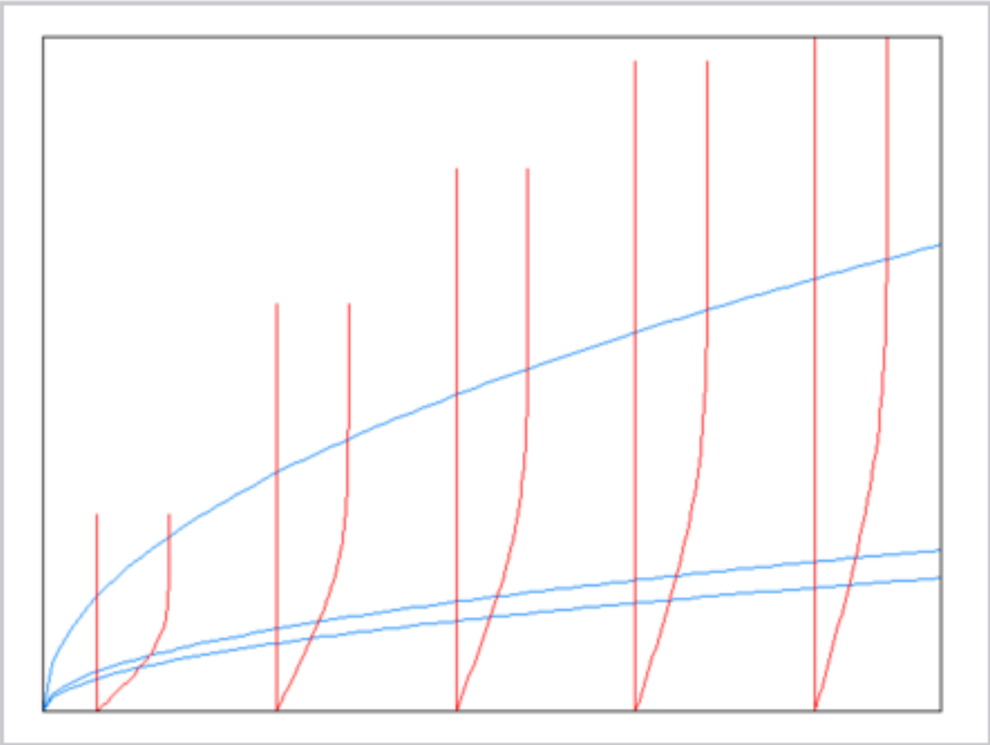
In [14]:
```
fig,ax=plt.subplots(1,3)
fig.set_figwidth(15)
fig.set_figheight(3)

for y in np.linspace(0,0.8,30):
    ax[0].arrow(1000,y,u(1000,y,U0=5),v(1000,y,U0=5))
    ax[1].arrow(4000,y,u(4000,y,U0=5),v(4000,y,U0=5))
    for x in [100,1000,2000,3000,4000,5000]:
        ax[2].arrow(x,y,u(x,y,U0=5)*100,v(x,y,U0=5)*100)


ax[0].set_title('(u,v) for x=1000')
ax[1].set_title('(u,v) for x=4000')
ax[2].set_title('(u,v) for different x')
ax[0].set_xlabel('x')
ax[1].set_xlabel('x')
ax[2].set_xlabel('x')
ax[0].set_ylabel('y')
ax[1].set_ylabel('y')
ax[2].set_ylabel('y')
plt.show()
```



We see that the third picture looks exactly as on the Wikipedia page for Blasius solution!

In [40]:
```
display.Image('blasius_wikipedia.png',width=500,height=500)
```

Out[40]:



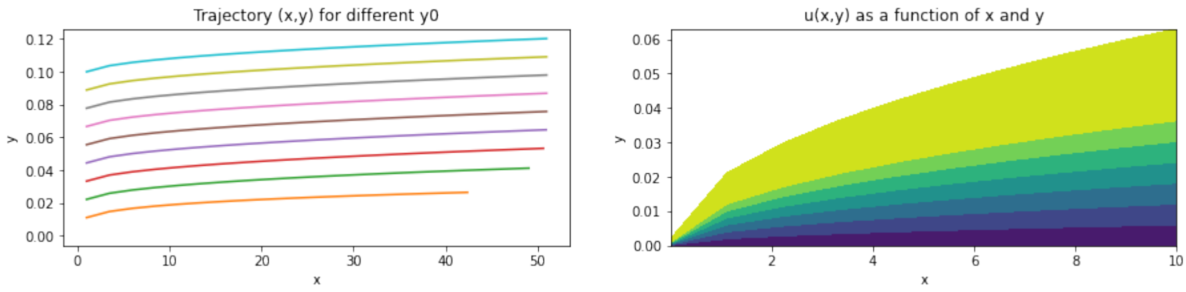
Developing Blasius boundary layer (not to scale). The velocity profile f' is shown in red at selected positions along the plate. The blue lines represent, in top to bottom order, the 99% free stream velocity line ($\delta_{99\%}$, $\eta \approx 5.29$), the displacement thickness (δ_* , $\eta \approx 1.79$) and $\delta(x)$ ($\eta = 1.51$). See Boundary layer thickness for a more detailed explanation.

Now let' draw trajectories of the integted system and also the contourplot for $u(x, y)$ as a colour, depending on x and y.

In [19]:

```
fig,ax=plt.subplots(1,2)
fig.set_figwidth(15)
fig.set_figheight(3)

#trajectories
dt=0.5
for y0 in np.linspace(0,0.1,10):
    trajectoryx=[]
    trajectoryy=[]
    x=1
    y=y0
    trajectoryx.append(x)
    trajectoryy.append(y)
    for t in range(20):
        dx=u(x,y,U0=5)
        dy=v(x,y,U0=5)
        x+=dt*dx
        y+=dt*dy
        trajectoryx.append(x)
        trajectoryy.append(y)
    ax[0].plot(trajectoryx,trajectoryy)



#contourplot
mas_x=[]
mas_y=[]
mas_u=[]
mas_v=[]
for x in np.linspace(0.01,10,10):
    #print(x,delta(x,U0))
    mas_x.append([])
    mas_y.append([])
    for exp_eta in np.linspace(1,100000,100):
        eta=np.log(exp_eta)
        y=eta*delta(x,U0=5)
        utek=u(x,y,U0=5)
        vtek=v(x,y,U0=5)
        mas_x[-1].append(x)
        mas_y[-1].append(y)
        mas_u.append(utek)
        mas_v.append(vtek)

X=mas_x
Y=mas_y
Z=np.array(mas_u).reshape(10,-1)

ax[1].contourf(X,Y,Z)

ax[0].set_title('Trajectory (x,y) for different y0')
ax[1].set_title('u(x,y) as a function of x and y')
ax[0].set_xlabel('x')
ax[1].set_xlabel('x')
ax[0].set_ylabel('y')
```

```
ax[0].set_ylabel('y')
ax[1].set_ylabel('y')
plt.show()
```



We see that close to the ground $u(x, y)$ is almost zero (because no slip condition), and then the wind speed increases up to $U_0 = 5$ meters per second.

Part B

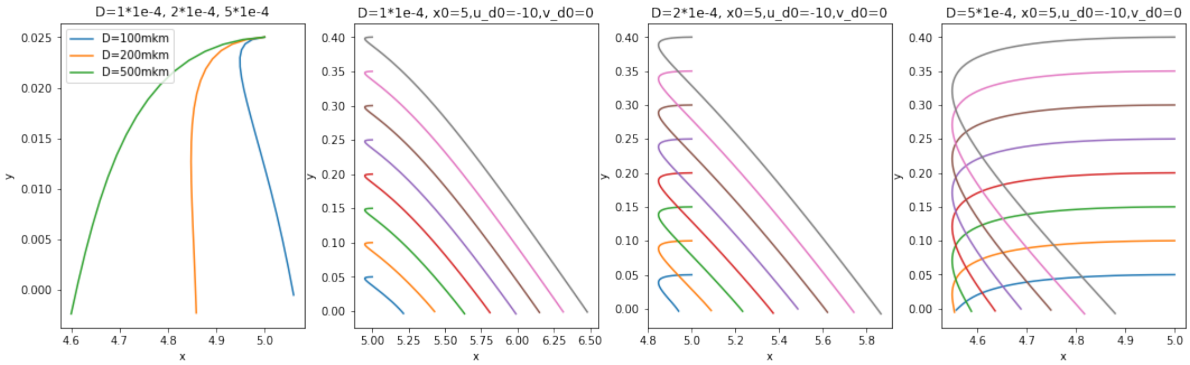
Let's integrate our system of ODE and see how trajectories with different initial y0 differ for drops with different diameters.

In [21]:

```
def get_trajectory(D,x0,y0,u_d0,v_d0,U0,mass_k=0):
    c_d=0.47
    rho_a=1.2255
    rho_d=1000
    g=9.8
    angle=1
    m_d0=rho_d*4/3*np.pi*D**3/8*(1-rho_a/rho_d)*angle


    def Fa(u_d,v_d,x,y,U0):
        u_a=u(x,y,U0)
        v_a=v(x,y,U0)
        return 0.5*c_d*rho_a*0.25*np.pi*D**2*np.sqrt((u_a-u_d)**2 +

    def Fgy(m_d):
        return -m_d*g

    def Fax(u_d,v_d,x,y,U0):
        return -Fa(u_d,v_d,x,y,U0)*(u_d-u_a)

    def Fay(u_d,v_d,x,y,U0):
        return -Fa(u_d,v_d,x,y,U0)*(v_d-v_a)


    dt=0.005
    trajectoryx=[]
    trajectoryy=[]
    x=x0
    y=y0
```

```
    y=y0
    u_d=u_d0
    v_d=v_d0
    trajectoryx.append(x)
    trajectoryy.append(y)
    i=0
    while y > 0:
        m_d=m_d0*np.exp(-mass_k*i*dt)
        i+=1
        #print(x,y)
        u_a=u(x,y,U0)
        v_a=v(x,y,U0)
        du_d=Fax(u_d,v_d,x,y,U0)/m_d
        dv_d=(Fay(u_d,v_d,x,y,U0)+Fgy(m_d))/m_d
        #print(x,y,u_a,v_a,du_d,dv_d)
        #print(Fay(u_d,v_d,x,y))
        #print(Fgy())
        u_d+=dt*du_d
        v_d+=dt*dv_d
        x+=dt*u_d
        y+=dt*v_d
        trajectoryx.append(x)
        trajectoryy.append(y)
    return trajectoryx,trajectoryy


fig,ax=plt.subplots(1,4)
fig.set_figwidth(18)
fig.set_figheight(5)


ax[0].set_title("D=1*1e-4, 2*1e-4, 5*1e-4")
tx,ty=get_trajectory(D=1*1e-4, x0=5,y0=0.025,u_d0=-10,v_d0=0,U0=5)
ax[0].plot(tx,ty, label='D=100mkm')
tx,ty=get_trajectory(D=2*1e-4, x0=5,y0=0.025,u_d0=-10,v_d0=0,U0=5)
ax[0].plot(tx,ty, label='D=200mkm')
tx,ty=get_trajectory(D=5*1e-4, x0=5,y0=0.025,u_d0=-10,v_d0=0,U0=5)
ax[0].plot(tx,ty, label='D=500mkm')
ax[0].legend()

ax[1].set_title("D=1*1e-4, x0=5,u_d0=-10,v_d0=0")
for y0 in np.linspace(0.05,0.4,8):
    tx,ty=get_trajectory(D=1*1e-4, x0=5,y0=y0,u_d0=-10,v_d0=0,U0=5)
    ax[1].plot(tx,ty)

ax[2].set_title("D=2*1e-4, x0=5,u_d0=-10,v_d0=0")
for y0 in np.linspace(0.05,0.4,8):
    tx,ty=get_trajectory(D=2*1e-4, x0=5,y0=y0,u_d0=-10,v_d0=0,U0=5)
    ax[2].plot(tx,ty)

ax[3].set_title("D=5*1e-4, x0=5,u_d0=-10,v_d0=0")
for y0 in np.linspace(0.05,0.4,8):
    tx,ty=get_trajectory(D=5*1e-4, x0=5,y0=y0,u_d0=-10,v_d0=0,U0=5)
    ax[3].plot(tx,ty)
```

```
ax[0].set_xlabel('x')
ax[1].set_xlabel('x')
ax[2].set_xlabel('x')
ax[3].set_xlabel('x')
ax[0].set_ylabel('y')
ax[1].set_ylabel('y')
ax[2].set_ylabel('y')
ax[3].set_ylabel('y')
plt.show()
```



On the first plot we see that that the greater the mass of the drop, the less is the impact of the wind on the trajectory. So in order to obtain desired behaviour of pesticide treatment it is better to use big droplets such as 300-400 mkm. And on the plots 2-4 we see that the impact of the hight of initial point on the trajectory also differs greatly when diameter differs. So in the next graph we will explore the impact of the height of inital point on the cone of trajectories for different wind speeds.

Recall that if the drop is released from the nozzle at speed SpeedFromNozzle with angle α to the vertical axis, then its horizontal component is SpeedFromNozzle$\cdot \sin\alpha$, and vertical component is SpeedFromNozzle$\cdot \cos\alpha$; From the nozzle drops are released at different angles and hence form the cone of trajectories. Let's draw this cone for different $y_0 = [0.5, 0.35, 0.2, 0.1]$ and different wind speeds = [0.1, 5,10,15] (that initiate different Blasius solutions).

I took drop exis velocity 20 meters per second based on velocities from Broumand article.

In [39]:
```
display.Image('drop_exit_velocity.png',width=600,height=600)
```

Out[39]:

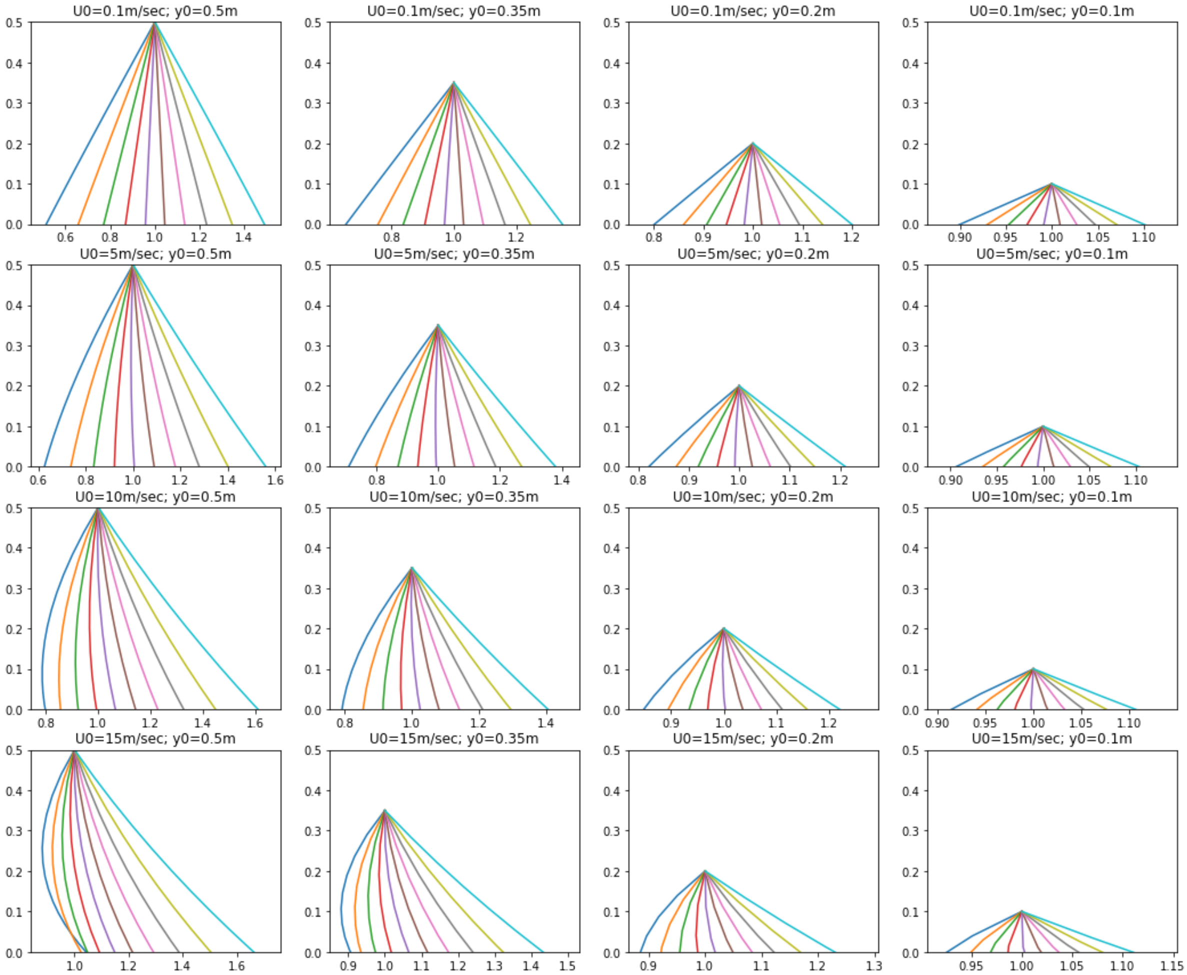
Physics of Fluids ARTICLE scitation.org/journal/phf

TABLE II. Experimental test conditions.

Case	Injection pressure, Δp (bar)	Sheet exit velocity, u_0 (m/s)[a]	Reynolds number based on u_0 (Re)	Weber number based on u_0 (We)	Ohnesorge number (Oh)[b]
1	2.1	18.7	6444	1677	0.0063
2	2.8	21.6	7441	2236	0.0063
3	4.8	28.6	9843	3912	0.0063
4	6.9	34.1	11 765	5589	0.0063
5	20.7	59.1	20 377	16 767	0.0063
6	34.5	76.3	26 307	27 945	0.0063
7	51.7	93.5	32 220	41 917	0.0063
8	68.9	107.9	37 204	55 889	0.0063

[a]Liquid sheet mean velocity at the nozzle exit is estimated based on $u_0 = 1.3\sqrt{\Delta p(\mathrm{kPa})}$ from Ref. 69.
[b]Viscous forces do not inhibit breakup as $Oh \sim 10^{-3}$

In [77]:
```
fig,ax=plt.subplots(4,4)
fig.set_figwidth(18)
fig.set_figheight(15)


wind_speeds=[0.1, 5,10,15]
y0s=[0.5,0.35,0.2,0.1]

SpeedFromNozzle=20

for i in range(len(wind_speeds)):
    U0=wind_speeds[i]
    for j in range(len(y0s)):
        y0=y0s[j]
        for alpha in np.linspace(-np.pi/4,np.pi/4,10):
            tx,ty=get_trajectory(D=5*1e-4, x0=1,y0=y0,u_d0=SpeedFro
            ax[i,j].plot(tx,ty)
            ax[i,j].set_ylim([0,0.5])
            ax[i,j].set_title("U0="+str(U0)+"m/sec; "+"y0="+str(y0)

plt.show()
```



We see two things. First, wind really matters and it may cause the cone to shift by 20 santimeters at speed 15 meters paer second; And since the nozzles are located on the beam of the tractor at such a distance that the cones intersect at the middle of their ways, that is 0.5 meters, then 0.2 meters of shift due to the wind will lead to sufficint interesection of the cones and hence overfertilization of the ground and addidtional costs on fertilizer itself. And the second fact is that the higher the initial point, the greater the wind affects the trajectory cone. So is will be beneficial to locate the beam of the tractor as low as possible (but not to harm the crops by pushing them by the beam).

Part C: add evaporation

We now will take into account the fact that due to evaporation the mass of the drop diminishes while it flies, and this fact means that the plants are reached by less amount of pesticide that left the fertilizer machine (tractor of unmanned aerial vehicle). In conditions of high wind (5-8-10 m/sec), high velocity of the tractor (10-15-20 m/sec), low relative humidity of the air (< 60%) and high temperature (>20 degrees of Celsius) up to 50% of water that left the machime may evaporate! And small drops (with diameter < 200 microns) will totally evaporate before they reach the ground! Coventry area is lucky because its average wind speed is about 5 m/sec, average relative humidity of the air varies betrween 60-96% and temperature varies between 10-20 degrees, so the effect of loosing fertilizer due to evaporation will not be as detrimetntal as 50%, but it may reach 10-20%, so we need to take this into account while modeling and also when we will calculate the estimated amount of fertilizer needed. So lets study how time to full evaporation depends on the diameter of the drop and choose which diameter is good for us.

In the agricular literature (attached, dissertation of Nadezhkina) we have the following formula to estimate total time to evaporation:

$$T = \frac{176.4d^2}{(1 + 1.92V_{wind})Deficit},$$

where V_{wind} is the speed of wind,

$T(min)$ is time to full evaporation in MINUTES,

$d(milimeters)$ is the diameter of the drop,

$Deficit(percents)$ is the deficit of stream elasticity.

For example, let's calculate time to full evaporation for a drop with diameter 100 microns = 0.1 milimeters, speed of wind 5m/sec and deficit of stream elasticity equal to 60%. And compare this time to an approximation of how much time would the drop need to reach the ground if there was no wind and no inital vertical speed and so the drop was simply falling down due to gravity: $h = \frac{gt^2}{2}$, so $t_{fall} = \sqrt{\frac{2h}{g}}$, and assume that the hight of our vehicle beam with nozzles is 0.5 meters.

In [79]:
```
d=0.1 #0.1mm=1e1-4=100microns
Vwind=5 #m/sec
Deficit=60 #persent
g=9.8
t_evap=176.4*d**2/((1+1.92*Vwind)*Deficit)*60  #sec
print("Time to full evaporation, sec:", t_evap)
h=0.5 #0.5m
t_fall=np.sqrt(2*h/g) #sec
print("Time to reach ground, sec:", t_fall)
```

```
Time to full evaporation, sec: 0.1664150943396227
Time to reach ground, sec: 0.3194382824999699
```

We see that for this diameter of drops time to fall is greater than time to evaporation! So no part of drop will reach the ground! So we need to increase the diameter of the drop to 200 microns and also it may be benefitial to decrease the hight of the beam with nozzles and make it 0.4 meters in order to allow the tractor to work when the wind is blowing (and the higher the beam is located, the greater the wind shifts the droplets, as we have seen in the pictures above).

In [80]:
```
d=0.2 #0.2mm=2e1-4=200microns
Vwind=5 #m/sec
Deficit=60 #persent
t_evap=176.4*d**2/((1+1.92*Vwind)*Deficit)*60  #sec
print("Time to full evaporation, sec:", t_evap)
h=0.5 #0.4m
t_fall=np.sqrt(2*h/g) #sec
print("Time to reach ground, sec:", t_fall)
```

```
Time to full evaporation, sec: 0.6656603773584908
Time to reach ground, sec: 0.3194382824999699
```

Now we see that the situation is better: some part of the drop will reach the ground. In fact, from the literature we can find that recommended drop size is 200-400 microns. Smaller drops will evaporate, and larger drops will not provide good and smooth coverage of the region with the fertilizer.

Now use the Mansurov's formula to estimate evaporation (in percents}:

$$E_{evap} = \frac{100 Deficit(1 + 1.92 V_{wind} t d^2)}{10584},$$

where Evap is the fraction of evaporated drop (in percents),

where

V_{wind} is the speed of wind,

$T(min)$ is time to full evaporation in seconds,

$d(milimeters)$ is the diameter of the drop,

$Deficit(percents)$ is the deficit of stream elasticity.

For example, if V_{wind} =2 m/sec, d=300 microns=0.3mm, $t = 3sec$, $deficit = 60$,

then $E_{evap} = 100 \cdot 60(1 + 1.92 \cdot 2) \cdot 3/10584 \cdot 0.3 \cdot 0.3 = 0.656$.

So more than half of a drop has evaporated!

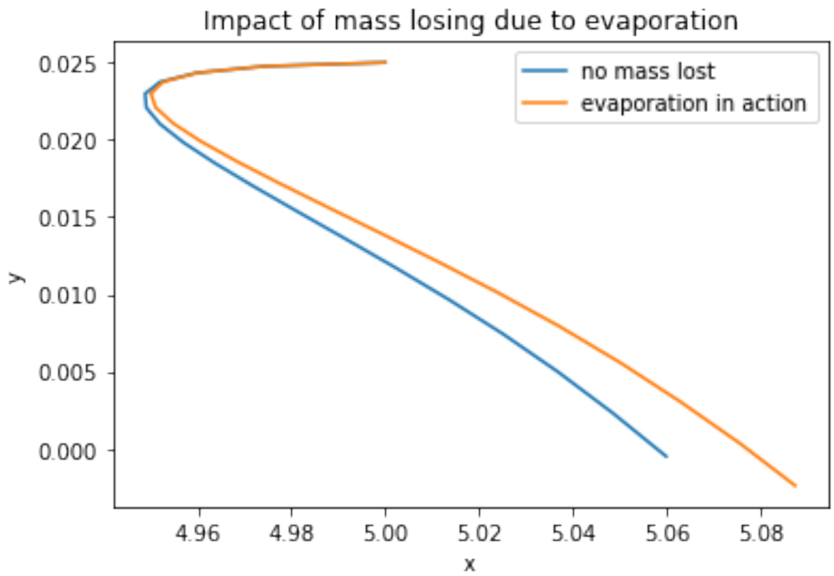
In [351]:
```
100*60*(1+1.92*2)*2.66/10584*0.3*0.3
```

Out[351]:
```
0.6568571428571429
```

Now lets introduce k such that $m(t) = m_0 e^{-kt}$, where coefficient k is responsible for evaporation and hence to drop mass reduction;

and let's take k such that the ground will be reached by a frop with $m = 0.7 m_0$ of initial drop.

In [27]:
```
k=-np.log(0.85)/2*50

tx,ty=get_trajectory(D=1*1e-4, x0=5,y0=0.025,u_d0=-10,v_d0=0,U0=5)
plt.plot(tx,ty,label='no mass lost')


txk,tyk=get_trajectory(D=1*1e-4, x0=5,y0=0.025,u_d0=-10,v_d0=0,U0=5
plt.plot(txk,tyk, label='evaporation in action')
plt.xlabel('x')
plt.ylabel('y')
plt.title('Impact of mass losing due to evaporation')
plt.legend()
plt.show()
```



We see that evaporation not only effects the amount of pecticide that reaches the ground but it also slightly changes the trajectory of the drops!

So drops become lighter and suffer more from shifts due to the wind.

Part D

Drop size distribution

As said in the provided in session 5 cambridge group report on properties of drops, the distribution of the drops is lognormal. I found the possible explanation of this emperical fact. Assuming that the appearance of different size in aerozol follows exponential distribution (this exponential distribution comes from exponential growth of instabilities in layers of liquid and streams preceeding the drop decay), we have that formula binding the change dD/D of drop size in diapason dD in aerosol with respect to its representative diameter D, depends on the constant speed of the growth of diversity of the drop size ϕ in the infinitesimally small time interval $d\tau$ as follows:

$$\frac{dD}{D} = \phi d\tau$$

; Here τ corresponds to moment t, normed by the characteristic time of destruction: $\tau = \frac{t}{t_b}$; When drops destruct, they break into smaller drops; Integrating the equation, we obtain that the normalized time interval, needed for creation of drops in the fixed class k of drop sizes is

$$\tau_k = \frac{\ln(D_k) - \ln(D_{n0.5})}{\phi}$$

, where $D_{n0.5}$ is a characteristic diameter with respect to range of sizes in aerozol, which in case of lognormal law is the median diameter (and in fertilizing machines they also write the median diameter of the drops!); After this we assume that the probability of addition of an infinitely small amount of drops into the class in the result of their size change is

$$\frac{dn_k}{N} = dp_k$$

, where N is the sample size, and dp_k is the infinitesimally small change of the value of frequency probability. And the last equation transforms this distribution of drop sizes by classes into a distribution of times. Due to the random nature of droplet formation, the distribution of times is normal. Therefore, assuming that the standard normal distribution, expresses the random nature of changes in the probability of each size class, the rate of probability change of a certain size class k is given by

$$\frac{dp_k}{d\tau} = \frac{1}{\sqrt{2\pi}} e^{-\frac{-\tau_k^2}{2}}$$

Using probability density function

$$f(D) = \frac{dp_k}{dD} = \frac{dp_k}{\phi D d\tau}$$

we result in lognormal function

$$f(D) = \frac{exp^{\frac{-(\ln(D/D_{n0.5}))^2}{2\phi^2}}}{\phi D\sqrt{2}}$$

They in the report also provided a formula by Dombrowski and Johns for the drop size:

$$d_D = \left[\frac{3\pi}{\sqrt{2}}\right]^{1/3} d_L \left[1 + \frac{3\mu}{(\rho\sigma d_L)^{1/2}}\right]^{1/6},$$

where d_D isn droplet diameter, μ is liquid's dunamic viscosity, ρ is the liquid's density, σ is the surface tension coefficient, and d_l is the ligament diameter,

$$d_L = 0.9614\left[\frac{K^2\sigma^2}{\rho_a \rho U^4}\right]^{1/6}\left[1 + 2.6\mu^3\sqrt{\frac{K\rho_a^4 U^7}{72\rho^2\sigma^5}}\right]^{1/5},$$

where K is the liquid sheet thickness times distance from the source, ρ_a is the density of the surrounding air, and U is the sheet velocity.

We have $\rho_a = 1\frac{kg}{m^3}$, $\rho = 1000\frac{kg}{m^3}$, $\sigma = 0.025\frac{N}{m}$, $\mu = 0.0009\frac{Pa}{sec}$, $U_0 = 20\frac{m}{sec}$, $D = 0.0001m$,

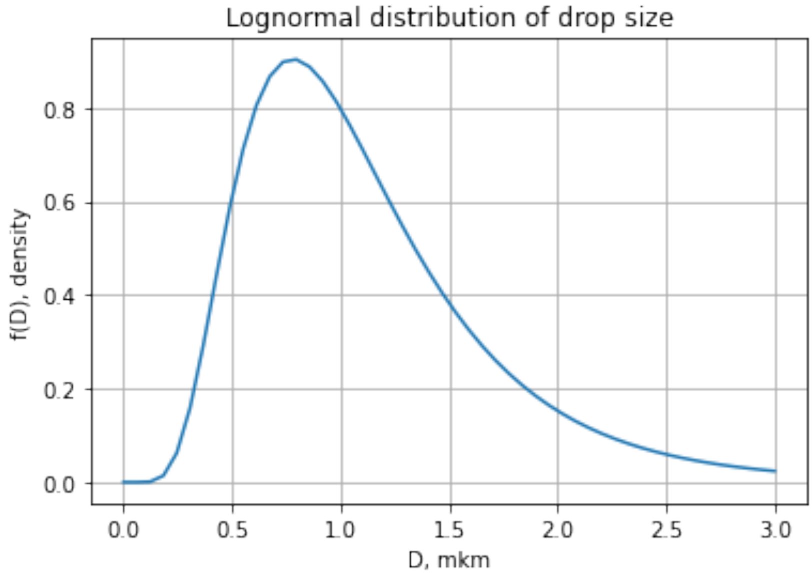
In [97]:
```
rho_a=1
rho=1000
sigma=0.025
mu=0.0009
U=20
K=0.0001
d_L=0.9614*((K**2*sigma**2)/(rho_a*rho*U**4))**(1/6)*(1+2.6*mu**3*n
d_D=(3*np.pi/np.sqrt(2))*(1/3)*d_L*(1+(3*mu)/(np.sqrt(rho*sigma*d_L
print("d_D=",d_D)
```

```
d_D= 0.0012487213213833465
```

We see $d_D = 12e - 4$, it is bigger than our $5e - 4$ used as droplet sized for fertilization, but is sounds not absurd.

Let's draw a sample distribution of a lognormal random variables which produces drop sizes from 100 miscrons to 300 miscrons (that is what we need for fertilization).

In [113]:
```
from scipy.stats import lognorm
mas_x=np.linspace(0,3)
plt.plot(mas_x, lognorm.pdf(mas_x, s=1/2,loc=0,scale=1))
plt.title('Lognormal distribution of drop size')
plt.grid()
plt.xlabel('D, mkm')
plt.ylabel('f(D), density')
plt.show()
```



And now let's present the distributions that are found in Cambridge report about drop size properties.

In [34]:
```
display.Image('drop_distrib1.png',width=500,height=500)
```

Out[34]:

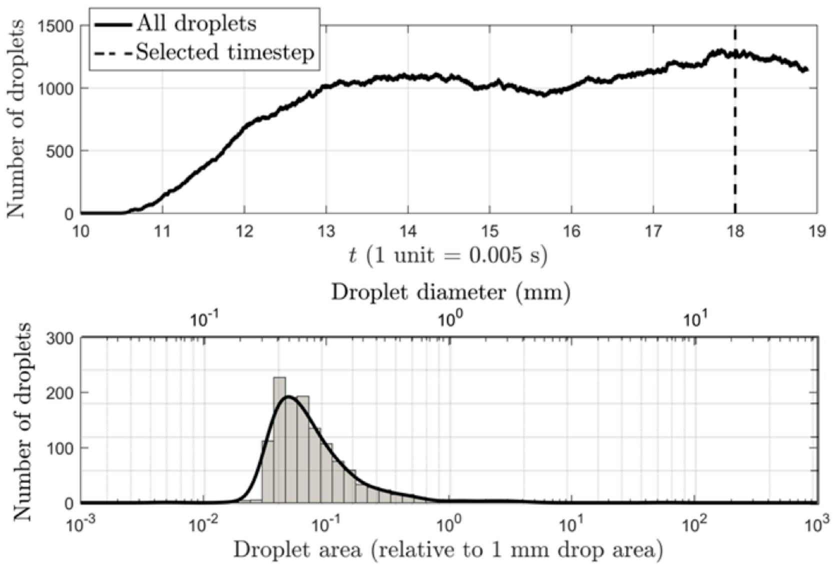


Figure 5: Example quantitative data obtained from a typical direct numerical simulation. Top: number of droplets in the computational domain in time, with a specific timestep selected in order to study the drop size distribution (bottom).

In [37]:
```
display.Image('drop_distrib2.png',width=900,height=700)
```

Out[37]:

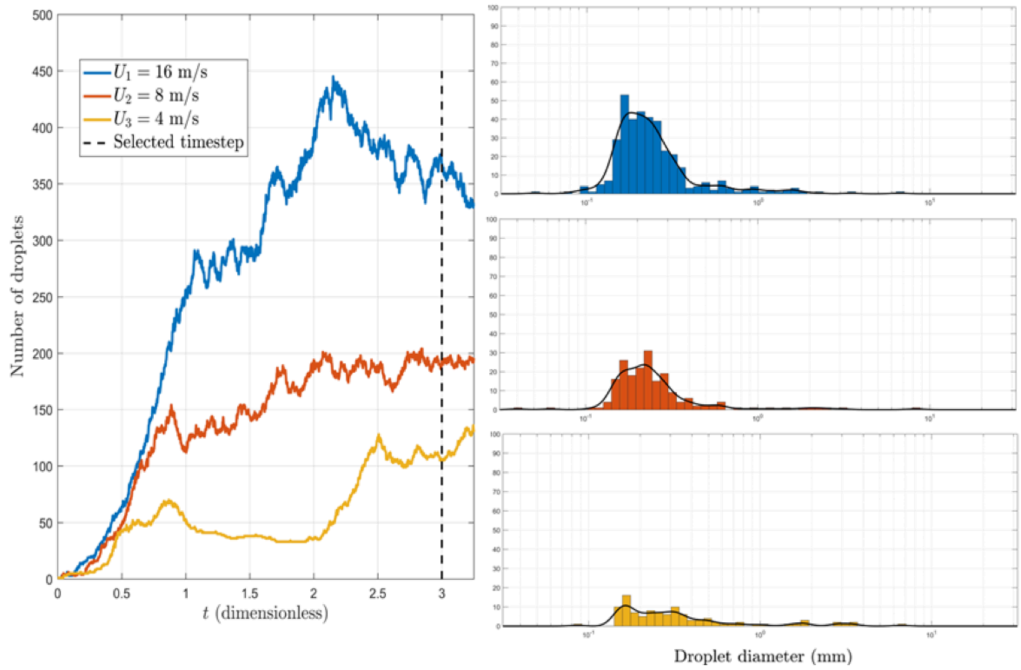


Figure 6: Varying velocity U from 4 m/s - 16 m/s: its effect on the number of droplets (left) and their distributions at a selected time (right).

Now let's discuss sutable pesticide concentration and composition, on the example of sunflowers. There are two kinds of pesticides that can be used when the sunflowers are alredy plants, not seeds: they are either those who make sunflowers grow bigger and have better seeds for future sunflower oil, or those that kill weeds but make no harm to sunflowers.

One of the most widespread pesticide against weeds that is used for sunflowers is called Beccard 150 KE; it contains metolachlor, terbutylazine, imazetapir, tribenuron-methyl, hizalofop-p-etil and quizalophop-p-tephuryl; and the active chemical is 125 grams per liter.

For making the crop better and richer, it is also worth considering enricing the soil with carbamide and ammonium nitrate. Typical percentage is 28%, but in 100 liters there will be 36 kg of fertilizer, because 100 liters of water weight 128 kg. We need to add water to use the fertilizer: 3 parts of water on one part of fertilizer.

We need to use 300 liters per hectar in order to have 40 droplets per squared santimeter; the field is 40 hectars; so we need 12000 liters; a pack of 10 liters of Baccard 125 costs 133 pounds; so we need 1200*133 =159.6 thousands of pounds to pay for each treatment of the field with this pesticide. If we take into account loosing pesticide due to evaportaion and shifting by the wind, we will need approximately 20% much fertilizer, so the cost wiil be 191.52k pounds.

In [41]:
```
display.Image('tractor.png',width=500,height=500)
```

Out[41]:

And now let's summarize all aspects that we found to be important if we want to use pesticides to fertilize crops in West Midlands.

0) Do not treat the filed in bad weather conditions! Wind greatet that 5-8 meters/sec, or temperature greater than 20 Celsius, or hunidity less that 60% is bad weather!

1) Use injector nozzles, not slot sprayers, because injector nozzles generate droplets of greater size which are therefore less prone to being shifted by wind, and therefor lead to less losses due to wind and also enable the tractor to work with (not very strong) wind. For example, if the hight of the nozzle is greater than 1 meter, the speed greater than 1 meter per second and the nozzle is not injector - than 50% of drops will not reach the ground!

2) Read carefully the techical instructions for the nozzles, understand that the color of the nozzle is responsible for the drop size. Use drop sizes as 200-400 miscrons, they are optimal for wind like 5 meters per second, as in West Midlands. 3)Not drive your tractor too fast, because there will be turbulent zone behind the tractor, and pesticinde will be attracted there, and there is risk for the crops behind the tractor to got burned by the pesticide.

4) Not neglect keeping your fertilizing equipment (a pomp, for example) in good condition and do nesessary renovation of equipment and nozzles, because old tired nozzles will give uneven coverage of the field.

5) Keep the beam with the nozzles at height 50-50 santimeters above the ground, if higher - the wind will have more effect as we have seen that the higher the nozzle is located, the more the wind shifts even big droplets.

6) Be ready that some percentage of droplets will evaporate - so you will need some more pesticide to treat the entire field than it may seem at first sight.

7) If possible buy a tractor that has compensator for vertical oscillations of the beam - because vertical oscillations affect very badly the trajectories of drops - and if the tractor is riding by a hummock, the konuses are colliding and the coverage of the field becomes uneven.

```
In [ ]:
```