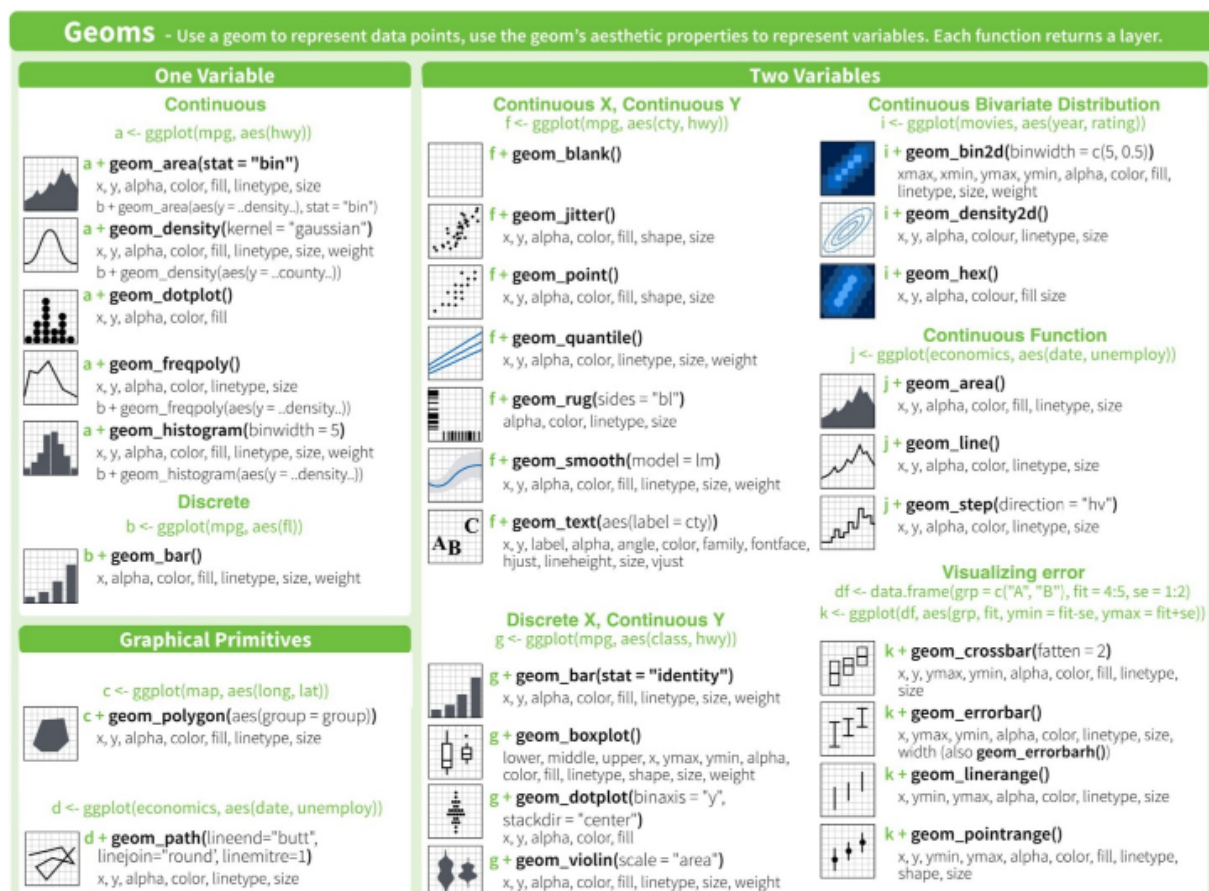


Unknown date

Unknown author

Large Collection of Neural Nets, Numpy, Pandas, Matplotlib, Scikit and ML Cheat Sheets

This collection covers much more than the topics listed in the title. It also features Azure, Python, Tensorflow, data visualization, and many other cheat sheets. Additional cheat sheets can be found [here](#) and [here](#). Below is a screenshot (extract from the data visualization cheat sheet.)



The one below is rather interesting too, but the source is unknown, and anywhere it was posted, it is unreadable. This is the best rendering after 30 minutes of work:

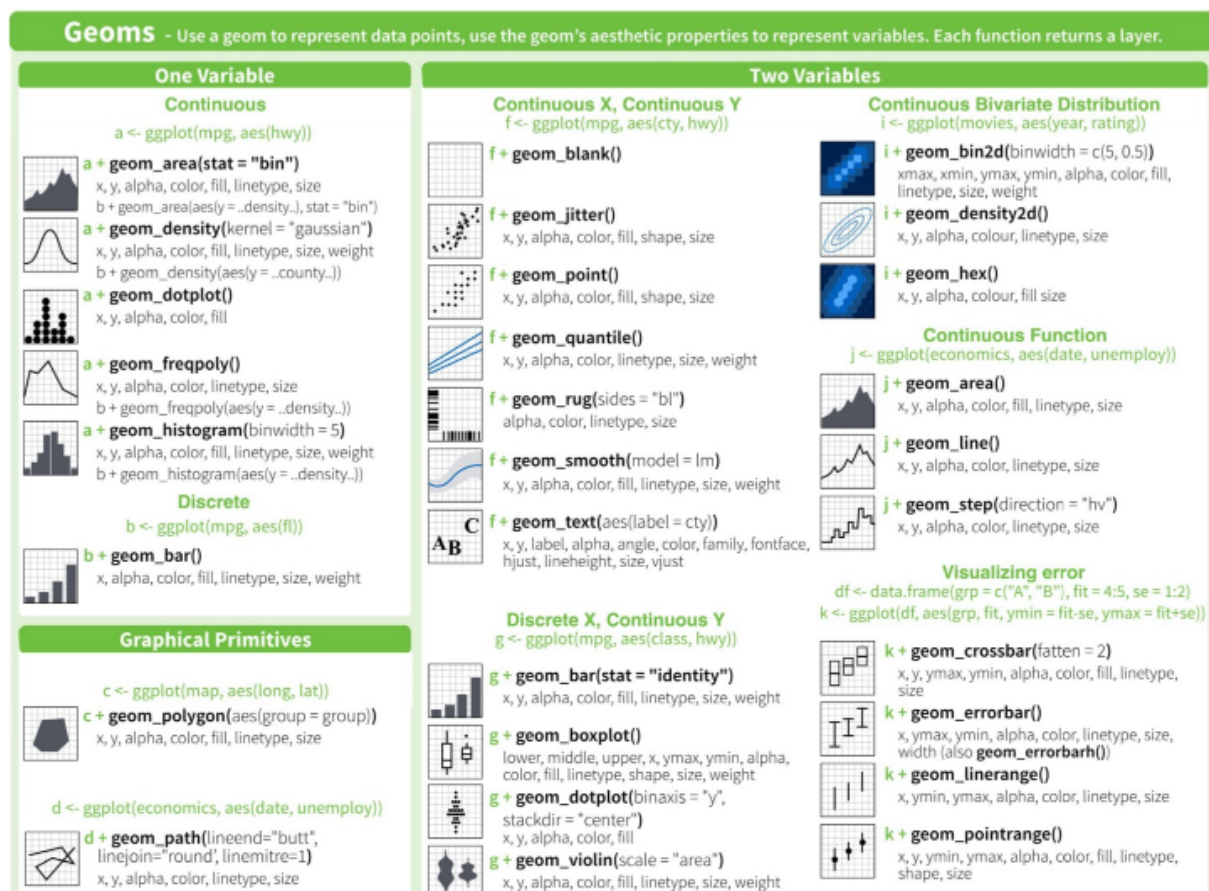
<p>General Minimization Algorithm:</p> $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k \text{ or } \Delta \mathbf{x}_k = (\mathbf{x}_{k+1} - \mathbf{x}_k) = \alpha_k \mathbf{p}_k$ <p>Steepest Descent Algorithm:</p> $\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k \mathbf{g}_k \text{ where, } \mathbf{g}_k = \nabla F(\mathbf{x}) _{\mathbf{x}=\mathbf{x}_k}$ <p>Stable Learning Rate: ($\alpha_k = \alpha$, constant) $\alpha < \frac{2}{\lambda_{\max}}$</p> <p>$\{\lambda_1, \lambda_2, \dots, \lambda_n\}$ Eigenvalues of Hessian matrix \mathbf{A}</p> <p>Learning Rate to Minimize Along the Line:</p> $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k \Rightarrow \alpha_k = -\frac{\mathbf{g}_k^T \mathbf{p}_k}{\mathbf{p}_k^T \mathbf{A} \mathbf{p}_k} \text{ (For quadratic fn.)}$ <p>After Minimization Along the Line:</p>	<p>*Heuristic Variations of Backpropagation:</p> <p>Batching: The parameters are updated only after the entire training set has been presented. The gradients calculated for each training example are averaged together to produce a more accurate estimate of the gradient. (If the training set is complete, i.e., covers all possible input/output pairs, then the gradient estimate will be exact.)</p> <p>Backpropagation with Momentum (MOBP):</p> $\Delta \mathbf{W}^m(k) = \gamma \Delta \mathbf{W}^m(k-1) - (1-\gamma) \alpha \mathbf{s}^m(\mathbf{a}^{m-1})^T$ $\Delta \mathbf{b}^m(k) = \gamma \Delta \mathbf{b}^m(k-1) - (1-\gamma) \alpha \mathbf{s}^m$ <p>Variable Learning Rate Backpropagation (VLBP)</p> <p>1. If the squared error (over the entire training set) increases by more than some set percentage ζ (typically one to five percent) after a weight update, then the weight update is discarded, the learning rate is multiplied by some factor $\rho < 1$, and the momentum coefficient γ is set to zero.</p>
--	--

Unknown date

Unknown author

Large Collection of Neural Nets, Numpy, Pandas, Matplotlib, Scikit and ML Cheat Sheets

This collection covers much more than the topics listed in the title. It also features Azure, Python, Tensorflow, data visualization, and many other cheat sheets. Additional cheat sheets can be found [here](#) and [here](#). Below is a screenshot (extract from the data visualization cheat sheet.)



The one below is rather interesting too, but the source is unknown, and anywhere it was posted, it is unreadable. This is the best rendering after 30 minutes of work:

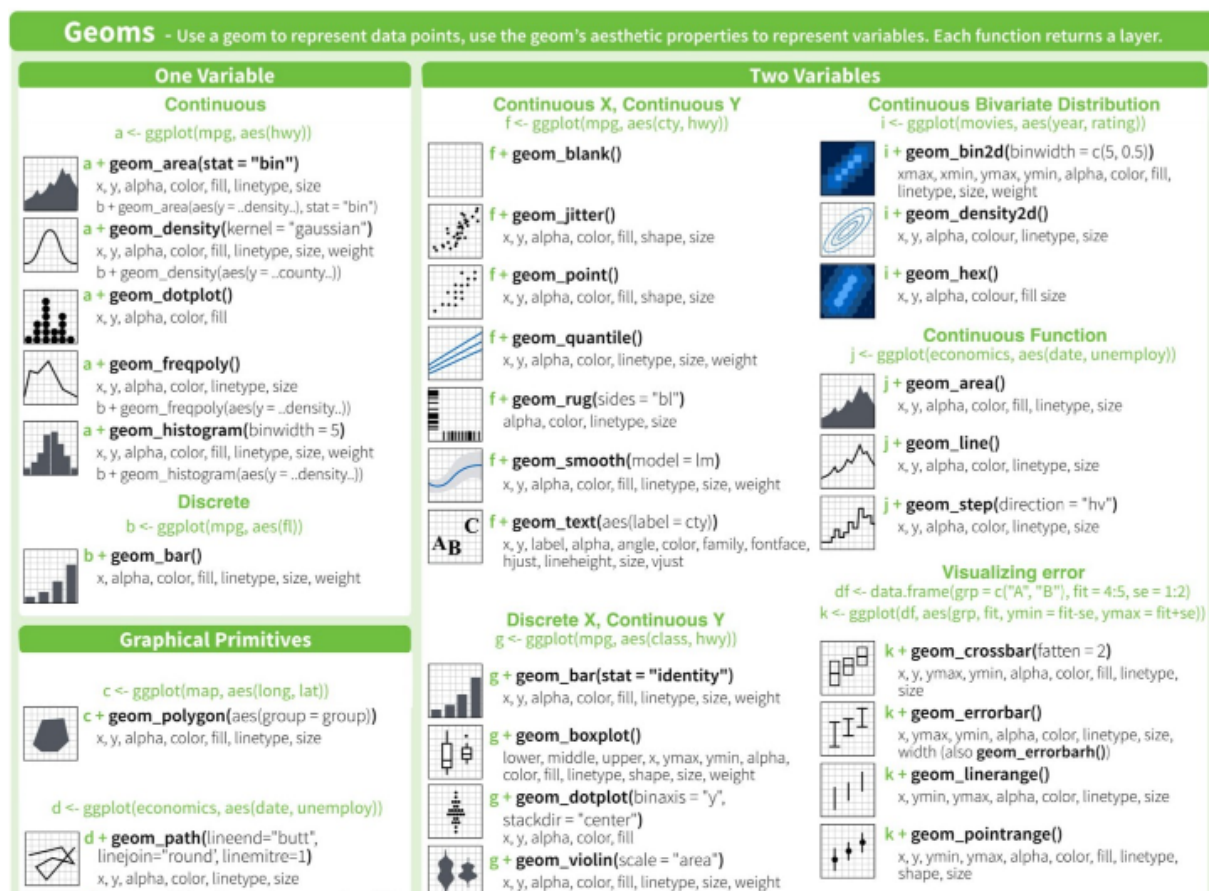
<p>General Minimization Algorithm:</p> $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k \text{ or } \Delta \mathbf{x}_k = (\mathbf{x}_{k+1} - \mathbf{x}_k) = \alpha_k \mathbf{p}_k$ <p>Steepest Descent Algorithm:</p> $\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k \mathbf{g}_k \text{ where, } \mathbf{g}_k = \nabla F(\mathbf{x}) _{\mathbf{x}=\mathbf{x}_k}$ <p>Stable Learning Rate: ($\alpha_k = \alpha$, constant) $\alpha < \frac{2}{\lambda_{\max}}$</p> <p>$\{\lambda_1, \lambda_2, \dots, \lambda_n\}$ Eigenvalues of Hessian matrix \mathbf{A}</p> <p>Learning Rate to Minimize Along the Line:</p> $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k \Rightarrow \alpha_k = -\frac{\mathbf{g}_k^T \mathbf{p}_k}{\mathbf{p}_k^T \mathbf{A} \mathbf{p}_k} \text{ (For quadratic fn.)}$ <p>After Minimization Along the Line:</p>	<p>*Heuristic Variations of Backpropagation:</p> <p>Batching: The parameters are updated only after the entire training set has been presented. The gradients calculated for each training example are averaged together to produce a more accurate estimate of the gradient. (If the training set is complete, i.e., covers all possible input/output pairs, then the gradient estimate will be exact.)</p> <p>Backpropagation with Momentum (MOBP):</p> $\Delta \mathbf{W}^m(k) = \gamma \Delta \mathbf{W}^m(k-1) - (1-\gamma) \alpha \mathbf{s}^m(\mathbf{a}^{m-1})^T$ $\Delta \mathbf{b}^m(k) = \gamma \Delta \mathbf{b}^m(k-1) - (1-\gamma) \alpha \mathbf{s}^m$ <p>Variable Learning Rate Backpropagation (VLBP)</p> <p>1. If the squared error (over the entire training set) increases by more than some set percentage ζ (typically one to five percent) after a weight update, then the weight update is discarded, the learning rate is multiplied by some factor $\rho < 1$, and the momentum coefficient γ is set to zero.</p>
--	--

Unknown date

Unknown author

Large Collection of Neural Nets, Numpy, Pandas, Matplotlib, Scikit and ML Cheat Sheets

This collection covers much more than the topics listed in the title. It also features Azure, Python, Tensorflow, data visualization, and many other cheat sheets. Additional cheat sheets can be found [here](#) and [here](#). Below is a screenshot (extract from the data visualization cheat sheet.)



The one below is rather interesting too, but the source is unknown, and anywhere it was posted, it is unreadable. This is the best rendering after 30 minutes of work:

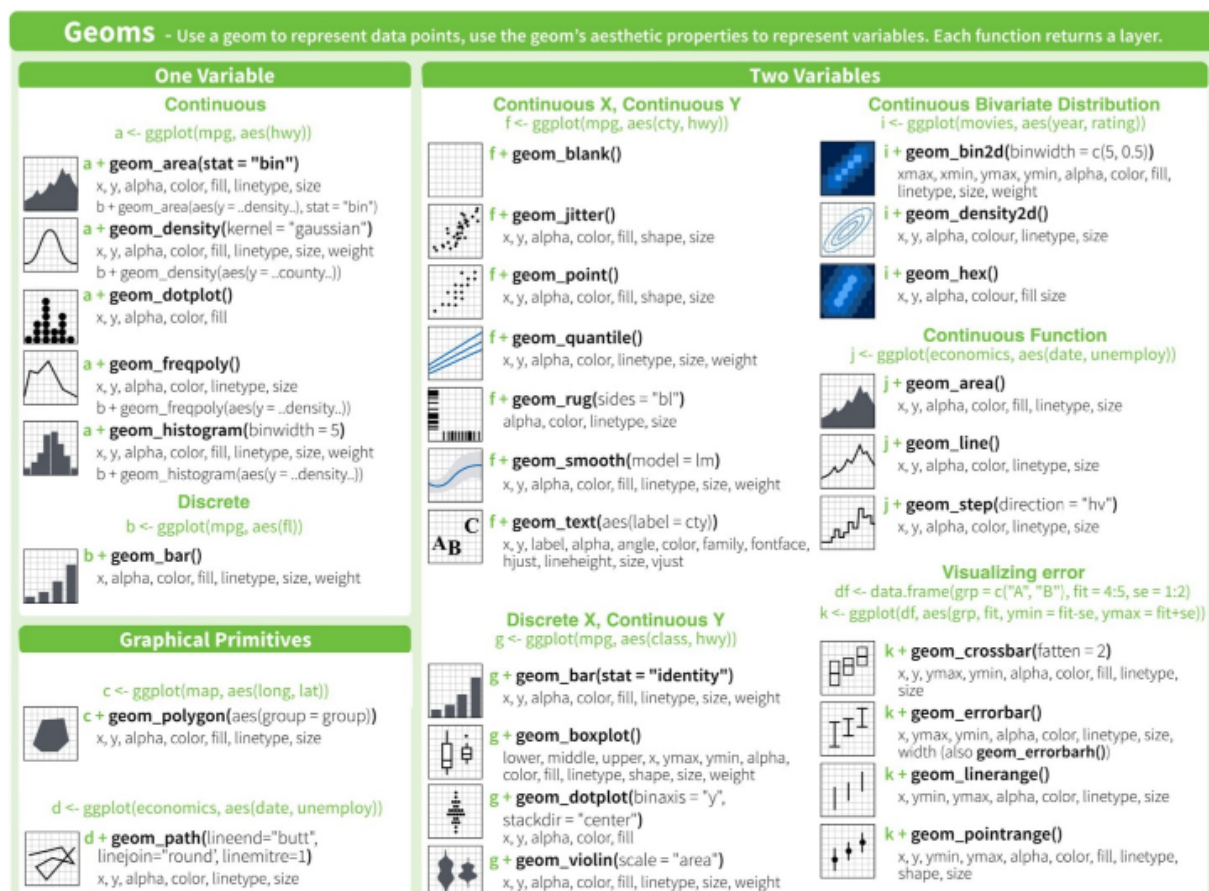
<p>General Minimization Algorithm:</p> $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k \text{ or } \Delta \mathbf{x}_k = (\mathbf{x}_{k+1} - \mathbf{x}_k) = \alpha_k \mathbf{p}_k$ <p>Steepest Descent Algorithm:</p> $\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k \mathbf{g}_k \text{ where, } \mathbf{g}_k = \nabla F(\mathbf{x}) _{\mathbf{x}=\mathbf{x}_k}$ <p>Stable Learning Rate: ($\alpha_k = \alpha$, constant) $\alpha < \frac{2}{\lambda_{\max}}$</p> <p>$\{\lambda_1, \lambda_2, \dots, \lambda_n\}$ Eigenvalues of Hessian matrix \mathbf{A}</p> <p>Learning Rate to Minimize Along the Line:</p> $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k \Rightarrow \alpha_k = -\frac{\mathbf{g}_k^T \mathbf{p}_k}{\mathbf{p}_k^T \mathbf{A} \mathbf{p}_k} \text{ (For quadratic fn.)}$ <p>After Minimization Along the Line:</p>	<p>*Heuristic Variations of Backpropagation:</p> <p>Batching: The parameters are updated only after the entire training set has been presented. The gradients calculated for each training example are averaged together to produce a more accurate estimate of the gradient. (If the training set is complete, i.e., covers all possible input/output pairs, then the gradient estimate will be exact.)</p> <p>Backpropagation with Momentum (MOBP):</p> $\Delta \mathbf{W}^m(k) = \gamma \Delta \mathbf{W}^m(k-1) - (1-\gamma) \alpha \mathbf{s}^m(\mathbf{a}^{m-1})^T$ $\Delta \mathbf{b}^m(k) = \gamma \Delta \mathbf{b}^m(k-1) - (1-\gamma) \alpha \mathbf{s}^m$ <p>Variable Learning Rate Backpropagation (VLBP)</p> <p>1. If the squared error (over the entire training set) increases by more than some set percentage ζ (typically one to five percent) after a weight update, then the weight update is discarded, the learning rate is multiplied by some factor $\rho < 1$, and the momentum coefficient γ is set to zero.</p>
--	--

Unknown date

Unknown author

Large Collection of Neural Nets, Numpy, Pandas, Matplotlib, Scikit and ML Cheat Sheets

This collection covers much more than the topics listed in the title. It also features Azure, Python, Tensorflow, data visualization, and many other cheat sheets. Additional cheat sheets can be found [here](#) and [here](#). Below is a screenshot (extract from the data visualization cheat sheet.)

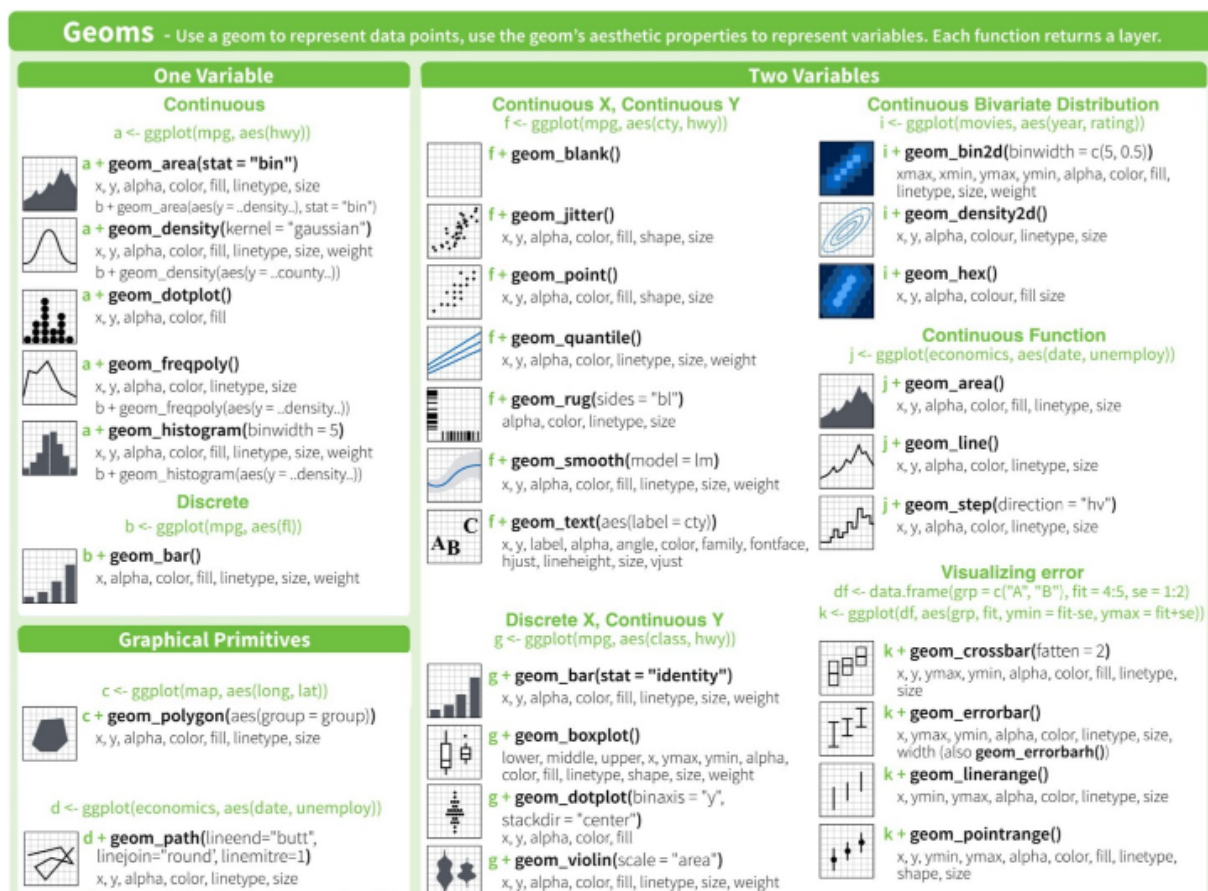


The one below is rather interesting too, but the source is unknown, and anywhere it was posted, it is unreadable. This is the best rendering after 30 minutes of work:

<p>General Minimization Algorithm:</p> $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k \text{ or } \Delta \mathbf{x}_k = (\mathbf{x}_{k+1} - \mathbf{x}_k) = \alpha_k \mathbf{p}_k$ <p>Steepest Descent Algorithm:</p> $\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k \mathbf{g}_k \text{ where, } \mathbf{g}_k = \nabla F(\mathbf{x}) _{\mathbf{x}=\mathbf{x}_k}$ <p>Stable Learning Rate: ($\alpha_k = \alpha$, constant) $\alpha < \frac{2}{\lambda_{\max}}$</p> <p>$\{\lambda_1, \lambda_2, \dots, \lambda_n\}$ Eigenvalues of Hessian matrix \mathbf{A}</p> <p>Learning Rate to Minimize Along the Line:</p> $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k \Rightarrow \alpha_k = -\frac{\mathbf{g}_k^T \mathbf{p}_k}{\mathbf{p}_k^T \mathbf{A} \mathbf{p}_k} \text{ (For quadratic fn.)}$ <p>After Minimization Along the Line:</p>	<p>*Heuristic Variations of Backpropagation:</p> <p>Batching: The parameters are updated only after the entire training set has been presented. The gradients calculated for each training example are averaged together to produce a more accurate estimate of the gradient. (If the training set is complete, i.e., covers all possible input/output pairs, then the gradient estimate will be exact.)</p> <p>Backpropagation with Momentum (MOBP):</p> $\Delta \mathbf{W}^m(k) = \gamma \Delta \mathbf{W}^m(k-1) - (1-\gamma) \alpha \mathbf{s}^m(\mathbf{a}^{m-1})^T$ $\Delta \mathbf{b}^m(k) = \gamma \Delta \mathbf{b}^m(k-1) - (1-\gamma) \alpha \mathbf{s}^m$ <p>Variable Learning Rate Backpropagation (VLBP)</p> <p>1. If the squared error (over the entire training set) increases by more than some set percentage ζ (typically one to five percent) after a weight update, then the weight update is discarded, the learning rate is multiplied by some factor $\rho < 1$, and the momentum coefficient γ is set to zero.</p>
--	--

Large Collection of Neural Nets, Numpy, Pandas, Matplotlib, Scikit and ML Cheat Sheets

This collection covers much more than the topics listed in the title. It also features Azure, Python, Tensorflow, data visualization, and many other cheat sheets. Additional cheat sheets can be found [here](#) and [here](#). Below is a screenshot (extract from the data visualization cheat sheet.)



The one below is rather interesting too, but the source is unknown, and anywhere it was posted, it is unreadable. This is the best rendering after 30 minutes of work:

<p>General Minimization Algorithm:</p> $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k \text{ or } \Delta \mathbf{x}_k = (\mathbf{x}_{k+1} - \mathbf{x}_k) = \alpha_k \mathbf{p}_k$ <p>Steepest Descent Algorithm:</p> $\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k \mathbf{g}_k \text{ where, } \mathbf{g}_k = \nabla F(\mathbf{x}) _{\mathbf{x}=\mathbf{x}_k}$ <p>Stable Learning Rate: ($\alpha_k = \alpha$, constant) $\alpha < \frac{2}{\lambda_{\max}}$</p> <p>$\{\lambda_1, \lambda_2, \dots, \lambda_n\}$ Eigenvalues of Hessian matrix \mathbf{A}</p> <p>Learning Rate to Minimize Along the Line:</p> $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k \Rightarrow \alpha_k = -\frac{\mathbf{g}_k^T \mathbf{p}_k}{\mathbf{p}_k^T \mathbf{A} \mathbf{p}_k} \text{ (For quadratic fn.)}$ <p>After Minimization Along the Line:</p>	<p>*Heuristic Variations of Backpropagation:</p> <p>Batching: The parameters are updated only after the entire training set has been presented. The gradients calculated for each training example are averaged together to produce a more accurate estimate of the gradient. (If the training set is complete, i.e., covers all possible input/output pairs, then the gradient estimate will be exact.)</p> <p>Backpropagation with Momentum (MOBP):</p> $\Delta \mathbf{W}^m(k) = \gamma \Delta \mathbf{W}^m(k-1) - (1-\gamma) \alpha \mathbf{s}^m(\mathbf{a}^{m-1})^T$ $\Delta \mathbf{b}^m(k) = \gamma \Delta \mathbf{b}^m(k-1) - (1-\gamma) \alpha \mathbf{s}^m$ <p>Variable Learning Rate Backpropagation (VLBP)</p> <p>1. If the squared error (over the entire training set) increases by more than some set percentage ζ (typically one to five percent) after a weight update, then the weight update is discarded, the learning rate is multiplied by some factor $\rho < 1$, and the momentum coefficient γ is set to zero.</p>
--	--