Parallelization of basic linear algebra operations

In the following, you don't have to program unless indicated explicitly.

1. How to parallelize the following operations:
   (a) $\underline{z} = \underline{x} + \alpha\, \underline{y}$ with $\alpha \in \mathbb{R}$
   (b) $\underline{y} = A\, \underline{x}$ where the matrix A is distributed column-wise

2. How to parallelize a Jacobi iteration for a fully populated matrix A? How does A have to be distributed? Which kind of difficulties do arise in the Gauss-Seidel method?

   Optional (programming): Write a parallel Jacobi iteration for the matrix $A = [a_{ij}]_{i,j=1}^{n}$ where $a_{ii} = \sum_{j \neq i} a_{ij}$ and a_{ij}, $i \neq j$ are randomly chosen (equally distributed in $[0,\,1]$ and pairwise independent).

3. How to parallelize the PCG method with the Jacobi preconditioner?

4. Consider the linear system $A\, \underline{x} = \underline{y}$ where the matrix A is tridiagonal and diagonal dominant.
   (a) Formulate Gauss' algorithm. How many operations are needed?
   (b) Find out the best parallelization of Gauss' algorithm. How should A be distributed?

5. Fast Fourier Transform (FFT):
   Let $\omega_n = e^{\frac{2i\pi}{n}} = \cos(\frac{2\pi}{n}) + i\, \sin(\frac{2\pi}{n})$ denote the n-th root of unity (here not normalized), and $F_n = \left[\omega_n^{(j-1)(k-1)}\right]_{j,k=1}^{n}$ the Fourier matrix of order n with $n = 2^l$.
   (a) Formulate the FFT algorithm which is able to perform the matrix-by-vector multiplication $F_n \underline{u}$ with only $\mathcal{O}(n \log_2 n)$ operations.
   (b) How to parallelize FFT?

6. Consider the function $f : \mathbb{R}^9 \to \mathbb{R}^9$:

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \end{bmatrix} \mapsto \begin{bmatrix} 100\, x_1 + x_2^2 + x_3^5 + x_8^4 \\ 100\, x_2 + x_4^2 + x_7^2 + x_6\, x_8 \\ 100\, x_3 + x_4 + x_5\, x_7 + x_9 \\ 100\, x_4 + x_1^5 + x_3^2\, x_6^4 + x_5^2\, x_7\, x_8 \\ 100\, x_5 + x_1 + x_7 + x_8\, x_9 \\ 100\, x_6 + x_2 + x_7\, x_8 + x_9 \\ 100\, x_7 + x_1 + x_2\, x_4\, x_8 + x_3 \\ 100\, x_8 + x_4^2 + x_5^2 + x_6\, x_7\, x_8 \\ 100\, x_9 + x_1^2 + x_3^6 + x_4^3 \end{bmatrix}$$

   How many operations are needed to compute components of $f(\vec{x})$, and what is the best way to distribute the evaluation of $f(\vec{x})$ among three processors of equal performance?